Complexity

Analysis and Applications of Location-Aware Big Complex Network Data

Lead Guest Editor: Jianxin Li
Guest Editors: Ke Deng, Xin Huang, and Jiajie Xu

Analysis and Applications of Location-Aware Big Complex Network Data

Complexity

Analysis and Applications of Location-Aware Big Complex Network Data

Lead Guest Editor: Jianxin Li
Guest Editors: Ke Deng, Xin Huang, and Jiajie Xu

Editorial Board

Contents

Hindawi
Complexity
Volume 2019, Article ID 3410262, 2 pages
https://doi.org/10.1155/2019/3410262

Editorial

Analysis and Applications of Location-Aware Big Complex Network Data

Jianxin Li,[1] **Ke Deng**,[2] **Xin Huang**,[3] **and Jiajie Xu**[4]

[1]*Deakin University, Australia*
[2]*RMIT University, Australia*
[3]*The Hong Kong Baptist University, Hong Kong, China*
[4]*Soochow University, China*

Correspondence should be addressed to Jianxin Li; jianxin.li@deakin.edu.au

Received 10 July 2019; Accepted 10 July 2019; Published 14 July 2019

In response to the ever-increasing challenges of location-aware network data like spatiosocial network and traffic network data, the network data processing technology is experiencing revolutionary changes in each stage including data collecting, cleaning, organizing, interpreting, analyzing, utilizing, and visualization. Those changes lead to a globally noticeable development trend of the convergence with big data frameworks, network analytical modeling, link or route prediction, and recommendation systems. This special issue aims at providing a forum to present recent advancements in the convergent research about big complex network data. Challenges include real-time event detection in a city, congestion discovery in a traffic network, location prediction of social users, social users' behavior recognition in physical world, and unified systems of processing multidimensional complex network data. The robust solutions call for highly innovative techniques in the fields including, but not limited to, machine learning, genetic algorithms, chaos, genetic algorithms, cellular automata, neural networks, and evolutionary game theory.

The special issue has attracted high-quality submissions from researchers worldwide in the areas of machine learning, artificial intelligence, data mining, natural language processing, data and web mining, and big data management to utilize their expertise and match up to the challenges of developing more efficient and practical algorithms or models to obtain smart knowledge from daily generated invaluable social network data and traffic network data. The total of submissions is 40. After single-blind peer review by at least two reviewers, 19 papers were finally accepted to be published. The accepted rate is 47.5%. The average number of authors for each accepted paper is 4.2. The affiliated institutes of authors are from China, Australia, France, India, Saudi Arabia, and Canada. These accepted papers can be organized in different groups. The focus of the first group of articles is location-based social network research. The paper titled "Targeted Influential Nodes Selection in Location-aware Social Networks" by S. Yang et al. presented an effective solution to identify the target based influential nodes in location-aware social networks. The paper titled "Discovering Travel Community for POI Recommendation on Location-Based Social Networks" by L. Tang et al. improved the community detection method for high-quality point-of-interest recommendation. The other three papers—"A Joint Deep Recommendation Framework for Location-Based Social Networks" by O. Tal and Y. Liu, "Optimal Proxy Selection for Socioeconomic Status Inference on Twitter" by J. L. Abitbol et al., and "Incremental Bilateral Preference Stable Planning over Event Based Social Networks" by B. Li et al.—invented the new machine learning technologies to explore the useful annotations for users and recommendation. The focus of the second group of articles is traffic based road network research. The papers titled "An Effective Algorithm for Video-Based Parking and Drop Event Detection" by G. Li et al., "Location-Aware Web Service Composition Based on the Mixture Rank of Web Services and Web Service Requests" by J. Lu et al., and "Predicting

Quality of Service via Leveraging Location Information" by L. Chen et al. provided the effective solution to discover the important location of delivering high-quality service in smart city environment. The other three papers titled "Semantic-Aware Top-k Multirequest Optimal Route" by S. Wang et al., "A Novel Index Method for K Nearest Object Query over Time-Dependent Road Networks" by Y. Yang et al., and "A Destination Prediction Network Based on Spatiotemporal Data for Bike-Sharing" by J. Jiang et al. proposed efficient algorithms and models to choose the optimal route planning. The focus of the third group of articles is deep learning. The papers titled "Sign Prediction on Unlabeled Social Networks Using Branch and Bound Optimized Transfer Learning" by W. Yuan et al., "Cognitive Driven Multilayer Self-Paced Learning with Misclassified Samples" by Q. Zhu et al., and "A Block Object Detection Method Based on Feature Fusion Networks for Autonomous Vehicles" by Q. Meng et al. investigated the deep learning technologies to optimize the entity recognition in social network data. The other five papers titled "pSPARQL: A Querying Language for Probabilistic RDF Data" by H. Fang, "Edge Computing in an IoT Base Station System: Reprogramming and Real-Time Tasks" by H. Wu et al., "Finding the Shortest Path with Vertex Constraint over Large Graphs" by Y. Yang et al., "Evaluation of Residential Housing Prices on the Internet: Data Pitfalls" by M. Li et al., and "Promoting Geospatial Service from Information to Knowledge with Spatiotemporal Semantics" by J. Geng et al. contributed a diverse range of topics using the semantic information. The solutions proposed in these research works include transfer learning, edge computing, data index structure, self-space learning, video analysis, feature fusion networks, multilayer classification, spatiotemporal pattern recognition, and probabilistic resource description framework data analysis. The effectiveness of the proposed solutions to the targeted problems has been reported based on empirical study and/or analysis.

In summary, the research papers cover a wide range of applications including geospatial services, location-based social network recommendation, social network influential node selection, data pitfall detection for residential house price prediction, socioeconomic status inference, parking detection, object detection for autonomous vehicle, service quality prediction, bike-sharing, label prediction in social networks, k nearest neighbor query over time-dependent road networks, and IoT base station system optimization.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work has been funded by the Australia Research Council Discovery Program under Grant No. DP160102114.

Jianxin Li
Ke Deng
Xin Huang
Jiajie Xu

Hindawi
Complexity
Volume 2019, Article ID 6059673, 15 pages
https://doi.org/10.1155/2019/6059673

Research Article

Optimal Proxy Selection for Socioeconomic Status Inference on Twitter

Jacob Levy Abitbol,[1] **Eric Fleury**,[2] **and Márton Karsai**[1]

[1]*Univ Lyon, Inria, CNRS, ENS de Lyon, Université Claude Bernard Lyon 1, LIP UMR 5668, F-69007 Lyon, France*
[2]*Inria, F-75012 Paris, France*

Correspondence should be addressed to Márton Karsai; marton.karsai@ens-lyon.fr

Received 25 January 2019; Accepted 15 April 2019; Published 19 May 2019

Guest Editor: Xin Huang

Individual socioeconomic status inference from online traces is a remarkably difficult task. While current methods commonly train predictive models on incomplete data by appending socioeconomic information of residential areas or professional occupation profiles, little attention has been paid to how well this information serves as a proxy for the individual demographic trait of interest when fed to a learning model. Here we address this question by proposing three different data collection and combination methods to first estimate and, in turn, infer the socioeconomic status of French Twitter users from their online semantics. We assess the validity of each proxy measure by analyzing the performance of our prediction pipeline when trained on these datasets. Despite having to rely on different user sets, we find that training our model on professional occupation provides better predictive performance than open census data or remote sensed expert annotation of habitual environments. Furthermore, we release the tools we developed in the hope it will provide a generalizable framework to estimate socioeconomic status of large numbers of Twitter users as well as contribute to the scientific discussion on social stratification and inequalities.

1. Introduction

Over the last decade the emergence of online social services changed the way we diffuse or acquire information and how we interact with each other. Every day billions of individuals use such services, while their penetration in our everyday lives seems ever-growing. In turn, online activities generate a massive volume of publicly available data, which are open to analysis, and fuel new data-driven developments in industry and research. These advances have led to the paradigm shift in the design of marketing strategies [1], emergence of new services, and open the door for data-driven reasoning on social phenomena relying on society-large observations [2]. The digital footprints left across these multiple media platforms provide us with a unique source to study patterns in human behavior down to the individual level, e.g., to understand how the linguistic phenotype of a given user is related to social attributes such as socioeconomic status (SES).

The quantification and inference of SES of individuals is a long-standing question in the social sciences. It is a rather difficult problem as it may depend on a combination of individual characteristics and environmental variables [3]. Some of these features can be easier assessed like income, gender, or age whereas others, relying to some degree on self-definition and sometimes entangled with privacy issues, are harder to assign like ethnicity, occupation, education level, or home location. Furthermore, individual SES correlates with other individual or network attributes, as users tend to build social links with others of similar SES, a phenomenon known as status homophily [4], arguably driving the observed stratification of society [5]. At the same time, shared social environment, similar education level, and social influence have been shown to jointly lead socioeconomic groups to exhibit stereotypical behavioral patterns, such as shared political opinion [6] or similar linguistic patterns [7]. Although these features are entangled and causal relation between them is far from understood, they appear as correlations in the data.

Datasets recording multiple characteristics of human behavior are more and more available due to recent developments in data collection technologies and increasingly popular online platforms and personal digital devices. The

automatic tracking of online activities (commonly associated with profile data and meta-information), the precise recording of interaction dynamics and mobility patterns collected through mobile personal devices, together with the detailed and expert annotated census data all provide new grounds for the inference of individual features or behavioral patterns [2]. The exploitation of these data sources has already been proven to be fruitful as cutting edge recommendation systems, advanced methods for health record analysis, or successful prediction tools for social behavior heavily rely on them [8]. Nevertheless, despite the available data, some inference tasks, like individual SES prediction, remain an open challenge.

The precise inference of SES would contribute to overcome multiple scientific challenges and could potentially have multiple commercial applications [9]. Further, robust SES inference would provide unique opportunities to gain deeper insights into socioeconomic inequalities [10], social stratification [5], and into the mechanisms driving network evolution, such as status homophily or social segregation.

In this work, we take a horizontal approach to this problem and explore various ways to infer the SES of a large sample of social media users. We propose different data collection and combination strategies using open, crawlable, or expert annotated socioeconomic data for the prediction task. Specifically, we use an extensive Twitter dataset of 1.3M users located in France, all associated with their tweets and profile information, 32,053 of them having inferred home locations. Individual SES is estimated by relying on three separate datasets, namely, socioeconomic census data, crawled profession information, and expert annotated Google Street View images of users' home locations. Each of these datasets is then used as ground-truth to infer the SES of Twitter users from profile and semantic features similar to [11]. We aim to explore and assess how the SES of social media users can be obtained and how much the inference problem depends on annotation and the user's individual and linguistic attributes. In addition, to demonstrate the power of our location inference method, we group users into nine distinct socioeconomic classes to identify correlations between the predictability of mobility [12] of geolocated users and their socioeconomic status. We observe that as a user's SES increases, so does his/her radius of gyration which in turn lowers the upper bound of predictability of his/her whereabouts.

We provide in Section 2 an overview of the related literature to contextualize the novelty of our work. In Section 3 we provide a detailed description of the data collection and combination methods including analysis of Twitter, census, mobility, occupation, and home location data. In Section 4 we introduce the features extracted to solve the SES inference problem, with results summarized in Section 5. Finally, in Sections 6 and 7 we conclude our paper with a brief discussion of the limitations and perspectives of our methods. Note that this paper is partially based on results published in a proceeding paper [13], while the source code of the reported methods has recently been released [14]. In terms of novelty, we outline the following contributions. (i) We introduced a new analytical framework to understand the relationship between mobility and socioeconomic status as well as its effect on our inference task; (ii) we provided a detailed analysis of the performance of our residential location filtering procedure; (iii) we delved into the study of our predictive performance, studying the set of features that were most determinant in inferring socioeconomic status; and finally (iv) we released to the scientific community all the pipelines used in this study to ease the collection and study of similar datasets as well as to motivate further study in this area.

2. Related Works

There is a growing effort in the field to combine online behavioral data with census records, and expert annotated information to infer social attributes of users of online services. The predicted attributes range from easily assessable individual characteristics such as age [15], or occupation [11, 16–18], to more complex psychological and sociological traits like political affiliation [19], personality [20], or SES [11, 21].

Predictive features proposed to infer the desired attributes are also numerous. In case of Twitter, user information can be publicly queried within the limits of the public API [22]. User characteristics collected in this way, such as profile features, tweeting behavior, social network, and linguistic content, have been used for prediction, while other inference methods relying on external data sources such as website traffic data [23] or census data [24, 25] have also proven effective. Nonetheless, only recent works involve user semantics in a broader context related to social networks, spatiotemporal information, and personal attributes [11, 17, 18, 26].

In this framework, aggregated studies of user spatial data have been particularly useful in fueling the analysis of human mobility patterns. Early work on mobile communication datasets [12, 27] showed that individuals tend to return to a few highly frequented locations leading to a high predictability in human travelling patterns. Analogous behavior was later exposed in Twitter [28], enabling the use of this social media platform as a proxy for tracking and predicting human movement.

Akin to this strand of research, user features collected from online platforms have also been used in the inference of individual demographic traits. The tradition of relating SES of individuals to their language dates back to the early stages of sociolinguistics where it was first shown that social status reflected through a person's occupation is a determinant factor in the way language is used [29]. This line of research was recently revisited by Lampos et al. to study the SES inference problem on Twitter. In a series of works [11, 17, 18, 26], the authors applied Gaussian Processes to predict user income, occupation, and socioeconomic class based on demographic, psycholinguistic features, and a standardized job classification taxonomy, which mapped Twitter users to their professional occupations. The high predictive performance has proven this concept with $r = 0.633$ for income prediction, and a precision of 55% for 9-ways SOC classification, and 82% for binary SES classification. Nevertheless, the models developed by the authors are learned by relying on datasets, which were manually labeled through an annotation

process crowdsourced through Amazon Mechanical Turk at a high monetary cost. Although the labeled data has been released and provides the base for new extensions [15], it has two potential shortfalls that need to be acknowledged. First, the method requires access to a detailed job taxonomy, in this case specific to England, which hinders potential extensions of this line of work to other languages and countries. Furthermore, the language to income pipeline seems to show some dependency on the sample of users that actively chose to disclose their profession in their Twitter profile. Features obtained on this set might not be easily recovered from a wider sample of Twitter users. This limits the generalization of these results without assuming a costly acquisition of a new dataset.

3. Data Collection and Combination

Our first motivation in this study was to overcome earlier limitations by exploring alternative data collection and combination methods. We study here three ways to estimate the SES of Twitter users by using (a) open census data, (b) crawled and manually annotated data on professional skills and occupation, and (c) expert annotated data on home location Street View images. We provide here a collection of procedures that enable interested researchers to introduce predictive performance and scalability considerations when interested in developing language to SES inference pipelines. In the following we present in detail all of our data collection and combination methods.

3.1. Twitter Corpus. Our central dataset was collected from Twitter, an online news and social networking service. Through Twitter, users can post and interact by "tweeting" messages with restricted length. Tweets may come with several types of metadata including information about the author's profile and the detected language as well as where and when the tweet was posted. Specifically, we recorded 90,369,215 tweets written in French, posted by 1.3 Million users in the time zones GMT and GMT+1 over one year (between August 2014 and July 2015) (the collection of this dataset was approved by the Ethic Committee of the hosting academic institute of the authors.). These tweets were obtained via the Twitter Powertrack API provided by Datasift with an access rate of 15%. Using this dataset we built various other corpora.

3.1.1. Geolocated Users. To find users with a representative home location we followed the method published in [30, 31]. As a bottom line, we concentrated on 127,614 users who posted at least five geolocated tweets with valid GPS coordinates, with at least three of them within a valid census cell (for definition see later), and over a longer period than seven days. Applying these filters we obtained 1,000,064 locations from geolocated tweets. By focusing on the geolocated users, we kept those with limited mobility, i.e., with median distance between locations not greater than 30 km, with tweets posted at places and times, which did not require travel faster than 130 km/h (maximum speed allowed within France), and with no more than three tweets within a two seconds window. We further filtered out tweets with coordinates corresponding to locations referring to places (such as "Paris" or "France"). Thus, we removed locations that did not exactly correspond to GPS-tagged tweets and also users, which were most likely bots.

Home location was estimated by the most frequent location for a user among all coordinates she visited. This way we obtained 32,053 users, each associated with a unique home location. Finally, we collected the latest 3,200 tweets from the timeline of all of geolocated users using the Twitter public API [22]. Note that by applying these consecutive filters we obtained a more representative population as the Gini index, indicating overall socioeconomic inequalities, was 37.3% before filtering becoming 36.4% due to the filtering methods, which is closer to the value reported by the World Bank (33.7%) [32].

To verify our results, we computed the average weekday and weekend distance from each recorded location of a user to her inferred home location defined either as her most frequent location overall or among locations posted outside of work-hours from 9AM to 6PM (see Figures 1(a) and 1(b)). This circadian pattern displays great similarity to earlier results [31] with two maxima, roughly corresponding to times at the workplace, and a local minimum at 1PM due to people having lunch at home for locations posted in weekdays. Moreover, when comparing the weekday and weekend patterns of behavior, we notice that the average distance to the inferred home location seemed to be smaller when considering only geolocations posted during the weekend, most likely due to the absence of the home-to-work commuting during the weekend. We found that this circadian pattern was more consistent with earlier results [31] when we considered all geolocated tweets ("All" in Figure 1(a)) rather than only tweets including "home-related" expressions ("Night" in Figure 1(a)). To further verify the inferred home locations, for a subset of 29,389 users we looked for regular expressions in their tweets that were indicative of being at home [31], such as "chez moi", "bruit", "dormir" or "nuit". In Figure 1(c) we show the temporal distribution of the rate of the word "dormir" at the inferred home locations. This distribution appears with a peak around 10PM, which is very different from the overall distribution of geolocated tweets throughout the day considering any location (see Figure 1(b)).

3.1.2. Linguistic Data. To obtain meaningful linguistic data we preprocessed the incoming tweet streams in several ways. As our central question here deals with language semantics of individuals, retweets do not bring any additional information to our study, thus we removed them by default. We also removed any expressions considered to be semantically meaningless like URLs, emoticons, mentions of other users (denoted by the @ symbol), and hashtags (denoted by the # symbol) to simplify later postprocessing. In addition, as a last step of textual preprocessing, we downcased and stripped the punctuation from the text of every tweet.

3.2. Census Data. Our first method to associate SES to geolocated users builds on an open census income dataset at intraurban level for France [33]. Obtained from 2010

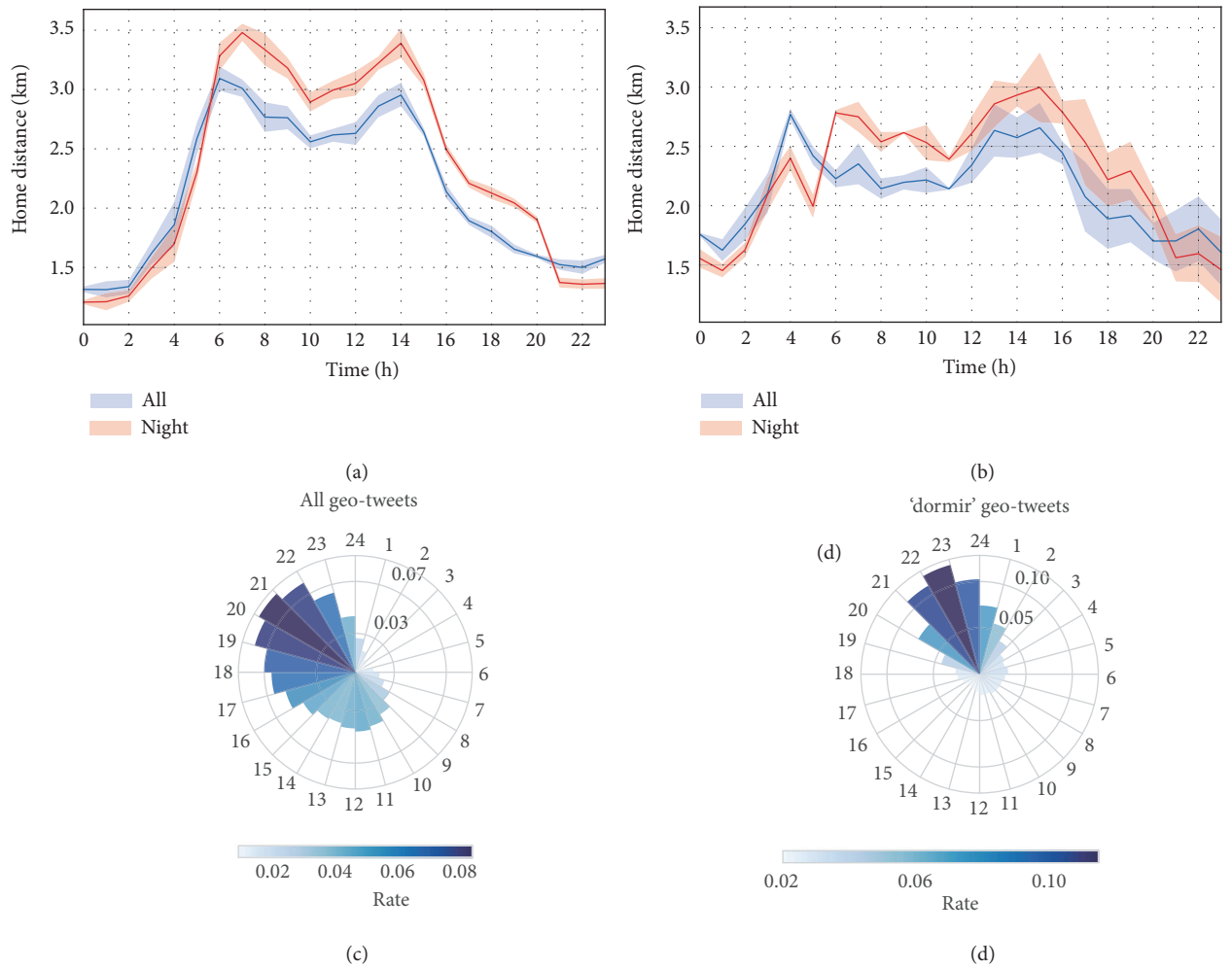


FIGURE 1: Average distance from home of active users per hour of the day depending on whether the geolocated tweet was posted during a weekday (a) or the weekend (b) and the most frequent location was chosen among all (blue) or only the night ones (red). (c) Hourly rate of all geolocated tweets and (d) geolocated tweets mentioning 'dormir' averaged over all weekdays.

French tax returns, it was released in December 2016 by the National Institute of Statistics and Economic Studies (INSEE) of France. This dataset collects detailed socioeconomic information of individuals at the census block level (called IRIS), which are defined as territorial cells with varying size but corresponding to blocks of around 2,000 inhabitants, as shown in Figure 2 for greater Paris. For each cell, the data records the deciles of the income distribution of inhabitants. Note that the IRIS data does not provide full coverage of the French territory, as some cells were not reported to avoid identification of individuals (in accordance with current privacy laws), or to avoid territorial cells of excessive area. Nevertheless, this limitation did not hinder our results significantly as we only considered users who posted at least three times from valid IRIS cells, as explained in Section 3.1.1.

To associate a single income value to each user, we identified the cell of their estimated home locations and assigned them with the median of the corresponding income distribution. Thus we obtained an average socioeconomic indicator for each user, which was distributed heterogeneously in accordance with Pareto's law [34]. This is demonstrated in Figure 6(a), where the $C(f)$ cumulative income distributions as the function of population fraction f appears as a Lorentz-curve with area under the diagonal proportional to socioeconomic inequalities. As an example, Figure 2 depicts the spatial distribution of 2,000 users with inferred home locations in IRIS cells located in central Paris and colored as the median income.

3.3. Mobility Analysis. In order to further assess the validity of the assignment of location to socioeconomic status, we studied the mobility traces generated by the set of geolocated users. Specifically, mirroring previous work [12, 28], we focused on the predictability of individual trajectories by analyzing the visitation patterns of the top $n = 10$ locations. To do so, given a user having visited at least $k = 5$ different census blocks, we computed $\pi_k(i)$ the fraction of time a user spends in the top i most visited location as

$$\pi_k(i) = \frac{N_i}{\sum_{j=1}^{n} N_j}, \quad i = 1, \ldots, n \tag{1}$$

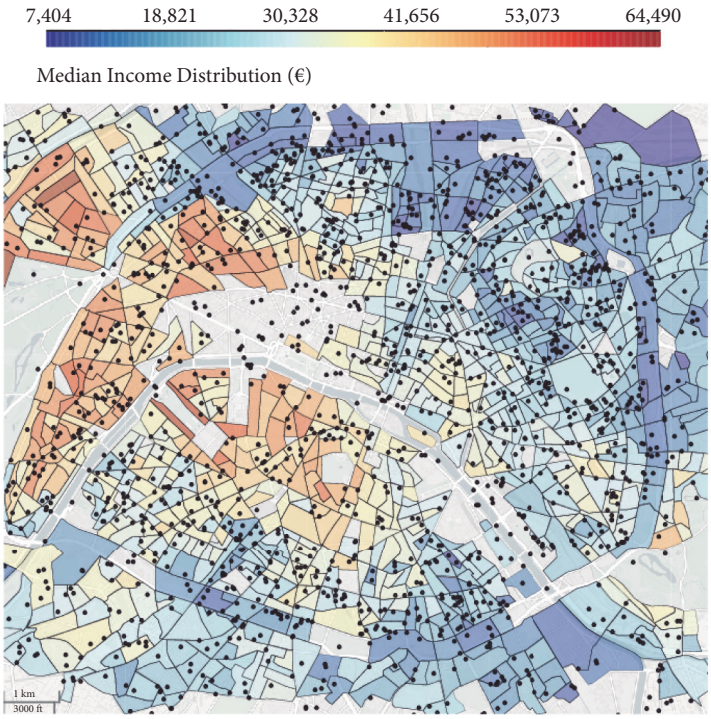


FIGURE 2: IRIS area cells in central Paris colored according to the median income of inhabitants, with inferred home locations of 2,000 Twitter users.

with N_i the number of times the user appeared in the i-th location. Note that in the above definition, $\forall i \leq k$, $N_i \in \mathbb{N}^*$. This metric was shown by Song et al. [12] to be an upper bound to an individual's predictability. At the same time, taking geolocated users' median income inferred from census cells we associated them into one of nine socioeconomic classes (1-poorest, 9-richest) following the social stratum model introduced in [5]. This procedure sorts users by their income, takes the CDF of income (shown in Figure 6(a)), and divides users into groups such that each group represents the same total sum of income. This partitioning, due to the Lorentz-curve shape of the CDF, provides socioeconomic classes with decreasing size with increasing income, as expected from theory and other real observations [5, 35].

As previously pointed out [12, 28] we noticed that, for all socioeconomic classes, the first and second most visited locations concentrate between 60 and 74% of all geolocations, suggesting the high likelihood of an individual tweeting preferentially from his/her home or office [28]. Furthermore, by aggregating users per socioeconomic class, an interesting trend was exposed: the higher the socioeconomic class in question, the lower the (upper bound of) predictability of the considered users was for any of the top n locations. Indeed, when studying how $\pi_5(1)$ is related to a user's socioeconomic status, we see that the higher one's socioeconomic class is, the less frequently he is seen at the most visited location (see Figure 3(a)). The robustness of this trend was further assessed by repeating our analysis for different values of k, always recovering this decreasing trend (see Figure 3(b)). The underlying explanation to this pattern may actually lie in the correlation between the socioeconomic status of people and the diversity of locations they visit. For instance, greater diversity may lead to a lower predictability of movement, which in turn might cause the observed trends. To control this, we computed the average radius of gyration r_g, describing the typical range of a user's trajectory per socioeconomic class, defined as

$$r_g = \sqrt{\frac{1}{L}\sum_{i=1}^{L}\left(\overrightarrow{r_i} - \overrightarrow{r_{cm}}\right)^2}. \quad (2)$$

Here $\overrightarrow{r_i}$ represents the position at time i, $\overrightarrow{r_{cm}} = (1/L)\sum_{i=1}^{L}\overrightarrow{r_i}$ is the center of mass of the trajectory, and L is the total number of recorded points for the user's location. As we see in Figure 3(c), r_g seems to increase on average as higher socioeconomic status is considered. Hence, high SES users tend to have more diverse mobility patterns than low SES ones, which in turn leads to lower predictability of their whereabouts. These results may relate to previous work [5, 36], which explain this trend by means of the positive payoff between commuting farther for better jobs, while keeping better housing conditions. As a consequence, the inferred home location for high SES users might be less precise due to their more dispersed mobility patterns and the lower predictability of their whereabouts. This is one reason why we define our SES inference later as a two-way inference problem, dividing users into a "rich" and a "poor" class.

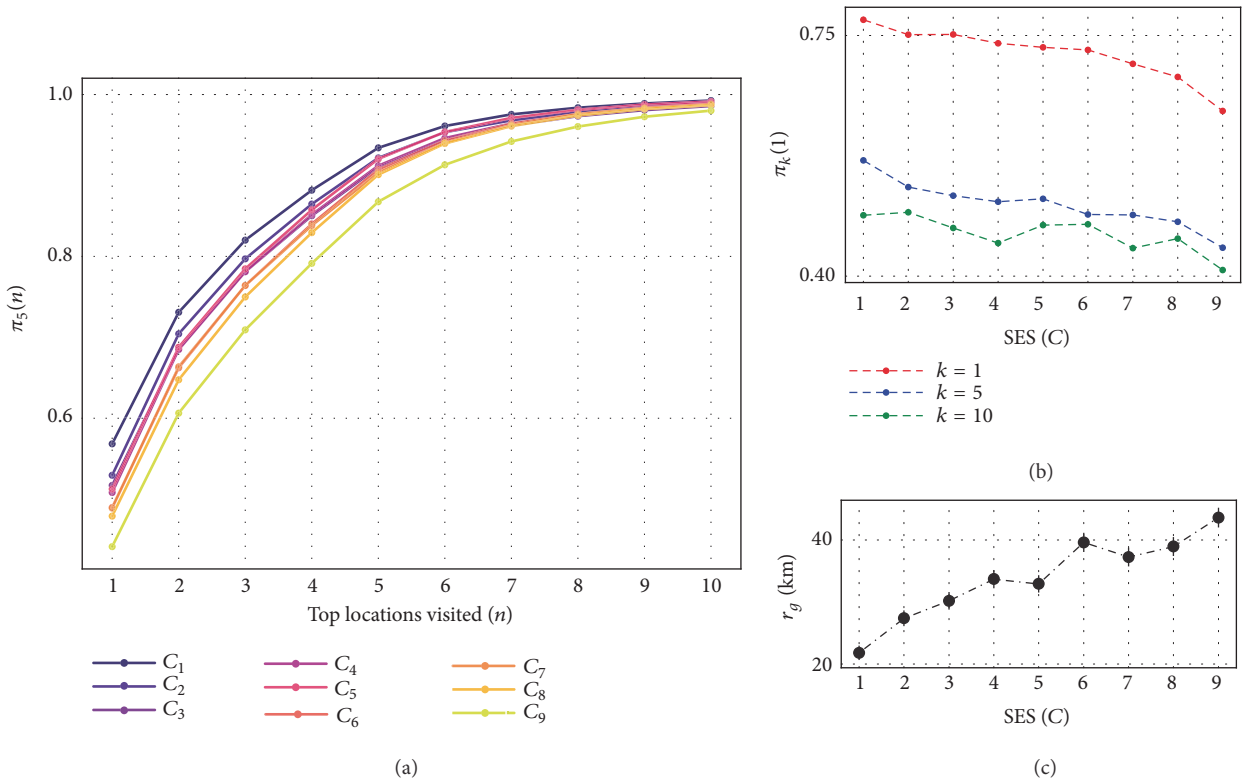


FIGURE 3: (a) The fraction of time a user spends in his/her top 10 most visited locations per socioeconomic class. (b) $\pi_k(1)$, fraction of time spent at the most visited location per socioeconomic class for users with at least k different locations. (c) Average radius of gyration per socioeconomic class (error bars are computed as $\sigma(r_g)/\sqrt{N}$, with N number of samples).

3.4. Occupation Data. Earlier studies [11, 17, 18] demonstrated that annotated occupation information can be effectively used to derive precise income for individuals and to infer therefore their SES. However, these methods required a somewhat selective set of Twitter users as well as an expensive annotation process by hiring premium annotators, e.g., from Amazon Mechanical Turk. Our goal here was to obtain the occupations for a general set of Twitter users without the involvement of annotators, but by collecting data from parallel online services.

As a second method to estimate SES, we took a sample of Twitter users who mentioned their LinkedIn [37] profile URL in their tweets or Twitter profile. Using these pointers we collected professional profile descriptions from LinkedIn by relying on an automatic crawler mainly used in Search Engine Optimization (SEO) tasks [38]. We obtained 4,140 Twitter/LinkedIn users all associated with their job title, professional skills, and profile description. Apart from the advantage of working with structured data, professional information extracted from LinkedIn is significantly more reliable than Twitter's due to the high degree of social scrutiny to which each profile is exposed [39].

To associate income to Twitter users with LinkedIn profiles, we matched them with a given salary based on their reported profession and an occupational-salary classification table provided by INSEE [40]. Due to the ambiguous naming of jobs and to acknowledge permanent/nonpermanent, senior/junior contract types we followed three strategies for the matching. In 40% of the cases we directly associated the reported job titles to regular expressions of an occupation. In 50% of the cases we used string sequencing methods [41] to associate reported and official names of occupations with at least 90% match. For the remaining 10% of users we directly inspected profiles. The distribution of estimated salaries reflects the expected income heterogeneities as shown in Figure 6(a). Users were eventually assigned to one of two SES classes based on whether their salary was higher or lower than the average value of the income distribution. Also note that LinkedIn users may not be representative of the whole population. We discuss this and other types of potential biases in Section 6.

3.5. Expert Annotated Home Location Data. Finally, motivated by recent remote sensing techniques, we sought to estimate SES via the analysis of the urban environment around the inferred home locations. Similar methodology has been lately reported by the remote sensing community [42] to predict sociodemographic features of a given neighborhood by analyzing Google Street View images to detect different car models, or to predict poverty rates across urban areas in Africa from satellite imagery [43]. Driven by this line of work, we estimated the SES of geolocated Twitter users as follows.

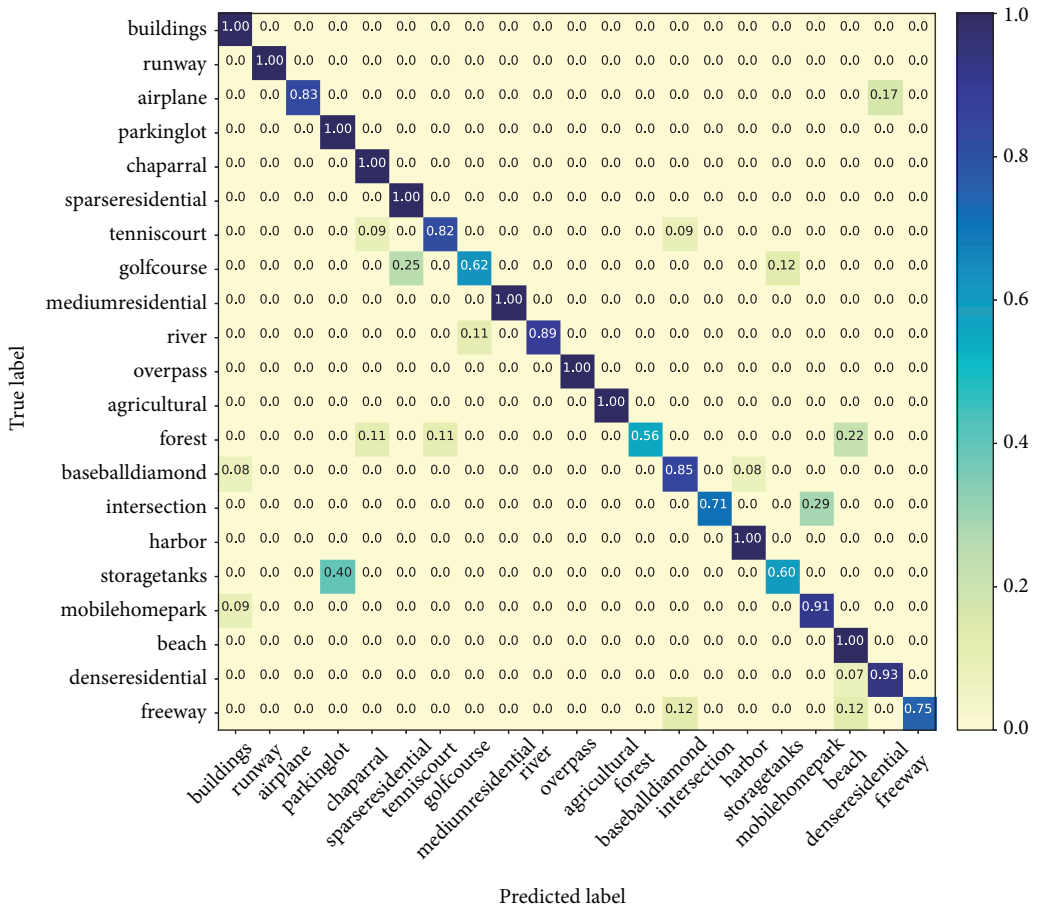


Figure 4: Confusion matrix in the test set obtained with a ResNet50 trained on the UCMerced for land use inference.

3.5.1. Preselection of Home Locations. Using geolocated users identified in Section 3.1.1, we further filtered them to obtain a smaller set of users with more precise inferred home locations. We screened all of their geotagged tweets and looked for regular expressions determining whether or not a tweet was sent from home [31]. As explained in Section 3.1.1, we exploited that "home-suspected" expressions appeared with a particular temporal distribution (see Figure 1(c)) since these expressions were used during the night when users are at home. This selection yielded 28,397 users mentioning "home-suspected" expressions regularly at their inferred home locations.

3.5.2. Identification of Urban/Residential Areas. In order to filter out inferred home locations not in urban/residential areas, we downloaded via Google Maps Static API [44] a satellite view in a $100m$ radius around each coordinate (for a sample see Figure 5(a)). To discriminate between residential and nonresidential areas, we built on land use classifier [45] using aerial imagery from the UC Merced dataset [46]. This dataset contains 2100 256×256 $1m/px$ aerial RGB images over 21 classes of different land uses (for a pair of sample images see Figure 5(b)). To classify land use, a CaffeNet architecture was trained, which reached accuracy over 95%. Here, we instantiated a *ResNet50* network using *keras* [47] pretrained on ImageNet [48], where all layers except the last five were frozen. The network was then trained with 10-fold cross-validation achieving a 93% accuracy after the first 100 epochs (cf. Figure 4). We used this model to classify images of the estimated home location satellite views (cf. Figure 5(a)) and kept those which were identified as residential areas (see Figure 5(b), showing the activation of the two first hidden layers of the trained model). This way, 5,396 inferred home locations were discarded.

3.5.3. Home Location Data with Expert Annotated SES. Next we aimed to estimate SES from architectural/urban features associated with the home locations. Given that our goal here was to lean on socioeconomic labels that were not estimated by the census, we forfeit the use of deep learning models to infer SES by relying upon human annotation. Thus, for each home location we collected two additional satellite views at different resolutions as well as six Street View images, each with a horizontal view of approximately 90°. We randomly selected a sample of 1,000 locations and involved architects to assign a SES score (from 1 to 9) to a sample set of selected locations based on the satellite and Street View around it (both samples had 333 overlapping locations). For validation, we took users from each annotated SES class and computed the distribution of their incomes inferred from the IRIS census data (see Section 3.2). Violin plots in Figure 5(d) show that in expert annotated data, as expected,

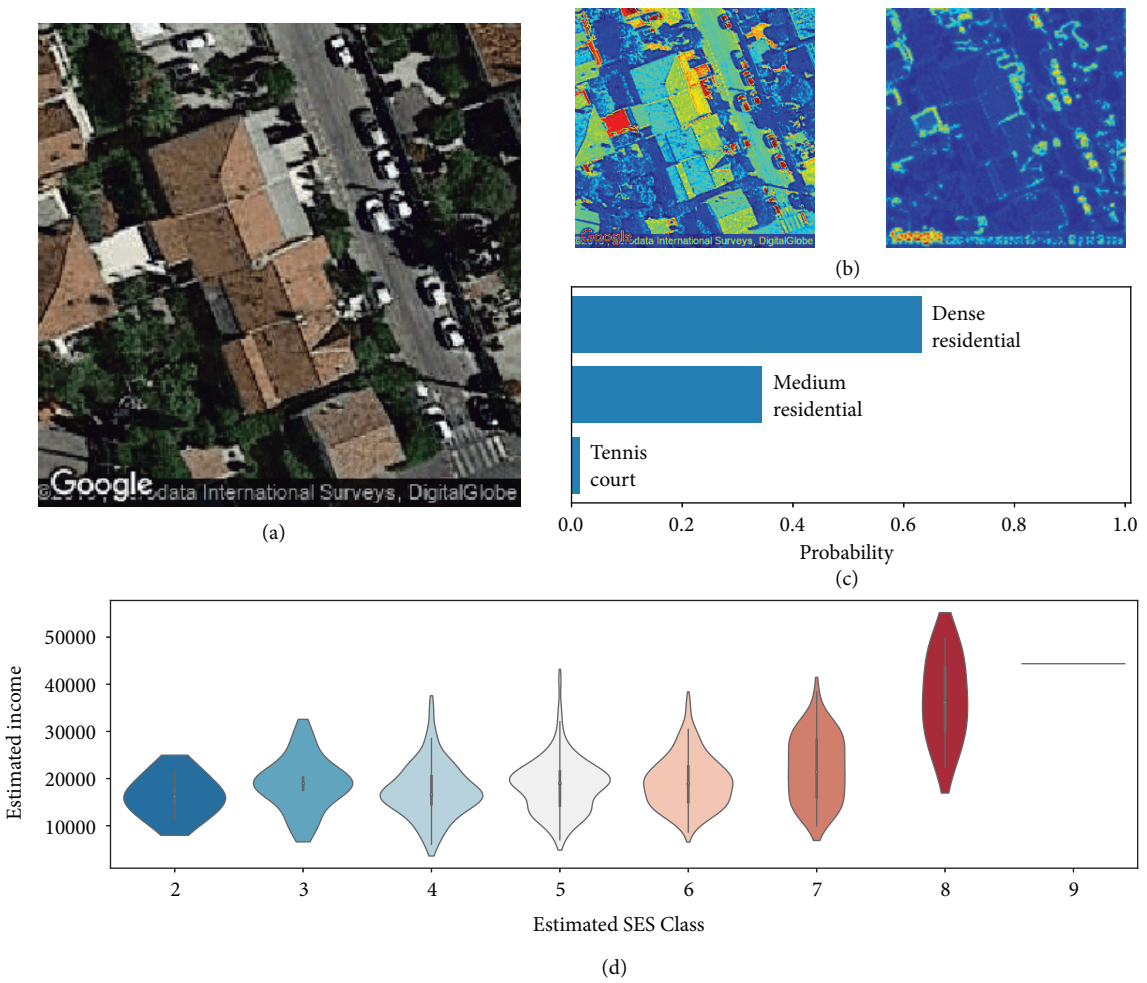


FIGURE 5: Top: ResNet50 Output: (a) original satellite view; (b) first two hidden layers activation; (c) final top-3 most frequent predicted area types; (d) architect SES score agreement with census median income for the sampled home locations. It is shown as violin plots of income distributions for users annotated in different classes (shown on x-axis and by color).

the inferred income values were positively correlated with the annotated SES classes. Labels were then categorized into two socioeconomic classes for comparison purposes. All in all, both annotators assigned the same label to the overlapping locations in 81.7% of samples.

To solve the SES inference problem we used the described three datasets (for a summary see Table 1). We defined the inference task as a two-way classification problem by dividing the user set of each dataset into two groups. For the census and occupation datasets the lower and higher SES classes were separated by the average income computed from the whole distribution, while in the case of the expert annotated data we assigned people from the lowest five SES labels to the lower SES class in the two-way task. The relative fractions of people assigned to the two classes are depicted in Figure 6(b) for each dataset and summarized in Table 1.

4. Feature Selection

Using the user profile information and tweets collected from every account's timeline, we built a feature set for each user, similar to Lampos et al. [11]. We categorized features into two sets, one containing shallow features directly observable from the data, while the other was obtained via a pipeline of data processing methods to capture semantic user features.

4.1. User Level Features. The user level features are based on the general user information or aggregated statistics about the tweets [17]. We therefore include general ordinal values such as the number and rate of retweets, mentions, and coarse-grained information about the social network of users (number of friends, followers, and ratio of friends to followers). Finally we vectorized each user's profile description and tweets and selected the top 450 and 560 1-grams and 2-grams, respectively, observed through their accounts (where the rank of a given 1-gram was estimated via *tf-idf* [49]).

4.2. Linguistic Features. To represent textual information, in addition to word count data, we used topic models to encode coarse-grained information on the content of the tweets of

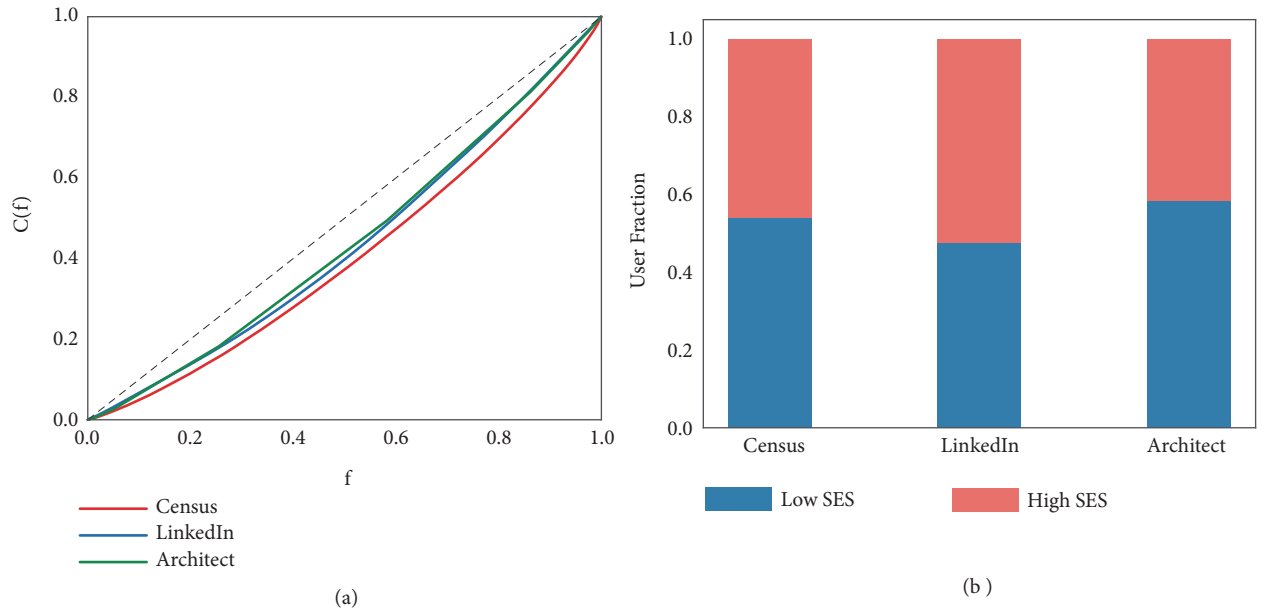


FIGURE 6: Cumulative distributions of income as a function of sorted fraction f of individuals. Dashed line corresponds to the perfectly balanced distribution. Distributions appear similar in spite of dealing with heterogeneous samples.

TABLE 1: Number of users and estimated fractions of low and high SES in each dataset.

	Census	Occupation	Expert
Size	32,053	4,140	1,000
Low SES	0.54	0.46	0.58
High SES	0.46	0.54	0.42

a user, similar to [11]. This enabled us to easily interpret the relation between semantic and socioeconomic features. Specifically, we started by training a *word2vec* model [50] on the whole set of tweets (obtained in the 2014-2015 timeframe) by using the skip-gram model and negative sampling with parameters similar to [15, 17]. To scale up the analysis, the number of dimensions for the embedding was kept at 50. This embedded words in the initial dataset in a $\mathbb{R}^{50}$ vector space.

Eventually we extracted conversation topics by running a spectral clustering algorithm on the word-to-word similarity matrix $M \in \mathbb{R}^{V\times V}$ with V vocabulary size and elements defined as the $M_{ij} = \langle u_i, u_j\rangle/\|u_i\|\|u_j\|$ cosine similarity between word vectors. Here $u_i \in \mathbb{R}^{50}$ is a vector of a word $i \in V$ in the embedding, $\langle\cdot\rangle$ is the dot product of vectors, and $\|\cdot\|$ is the L^2 norm of a vector. This definition allows for negative entries in the matrix to cluster, which were set to null in our case. This is consistent with the goal of the clustering procedure as negative similarities should not encode dissimilarity between pairs of words but orthogonality between the embeddings. This procedure was run for 50, 100, and 200 clusters and allowed the homogeneous distribution of words among clusters (hard clustering). The best results were obtained with 100 topics in the topic model. Finally, we manually labeled topics based on the words assigned to them and computed the topic-to-topic correlation matrix shown in Figure 7. There, after block diagonalization, we found clearly correlated groups of topics which could be associated with larger topical areas such as communication, advertisement, or soccer.

As a result we could compute a representative topic distribution for each user, defined as a vector of normalized usage frequency of words from each topic. Also note that the topic distribution for a given user was automatically obtained as it depends only on the set of tweets and the learned topic clusters without further parametrization.

To demonstrate how discriminative the identified topics were in terms of the SES of users we associated with each user the 9th decile value of the income distribution corresponding to the census block of their home location and computed for each labeled topic the average income of users depending on whether or not they mentioned the given topic. Results in Figure 8 demonstrate that topics related to politics, technology, or culture are more discussed by people with higher income, while other topics associated with slang, insults, or informal abbreviations are more used by people of lower income. These observable differences between the average income of people, who use (or not) words from discriminative topics, demonstrate well the potential of word topic clustering used as features for the inference of SES. All in all, each user in our dataset was assigned with a 1117 feature vector encoding the lexical and semantic profile she displayed

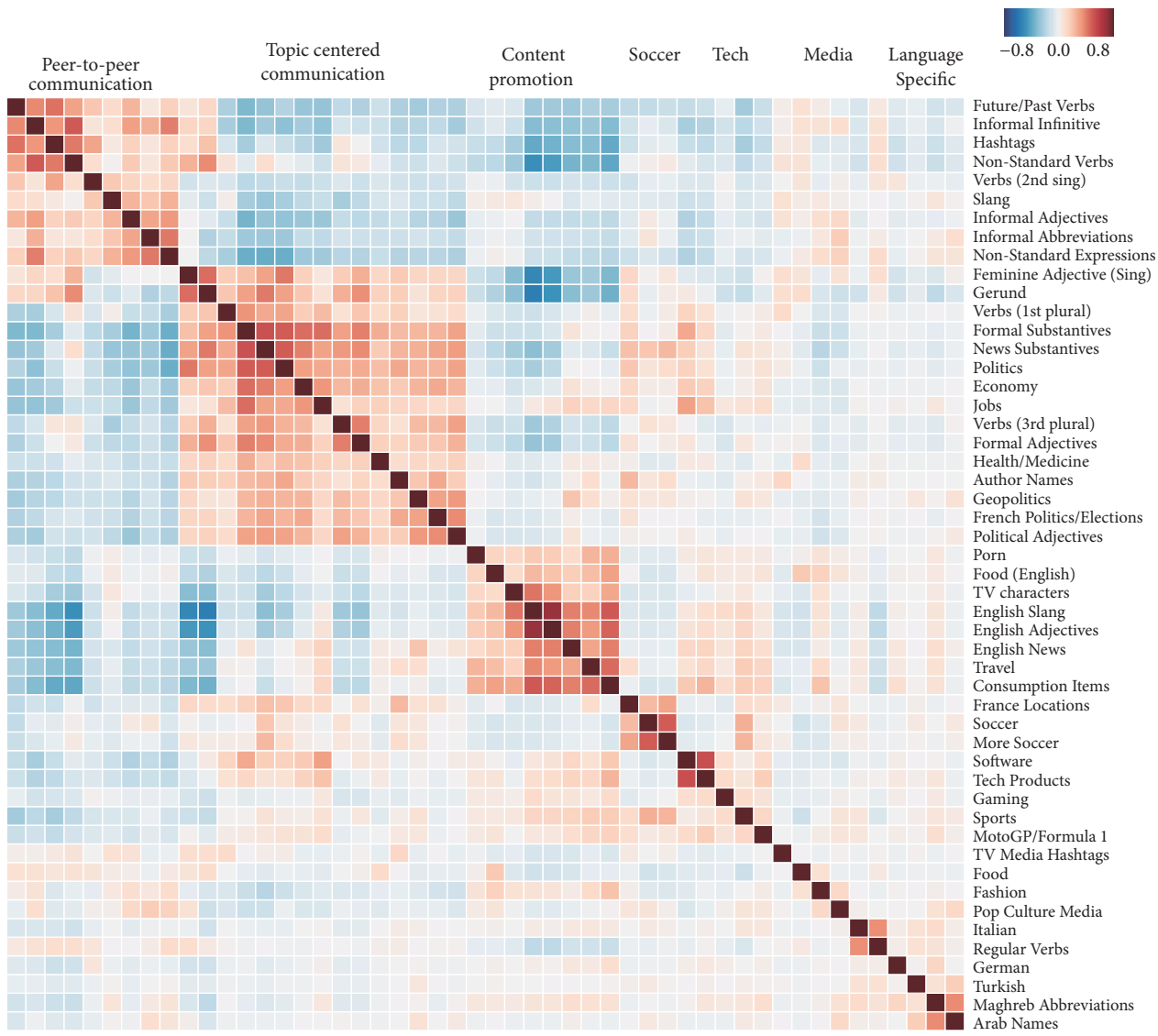


Figure 7: Clustered topic-to-topic correlation matrix: topics are generated via the spectral clustering of the word2vec word cosimilarity matrix. Row labels are the name of topics while column labels are their categories. Blue cells (resp. red) assign negative (resp. positive) Pearson's correlation coefficients.

on Twitter. We did not apply any further feature selection as the distribution of importance of features appeared rather smooth (not shown here). It did not provide evident ways to identify a clear set of particularly determinant features, but rather indicated that the combination of them was important.

5. Results

In order to assess the degree to which linguistic features can be used for discriminating users by their socioeconomic class, we trained with these feature sets different learning algorithms. Namely, we used the XGBoost algorithm [51], an implementation of the gradient-boosted decision trees for this task. Training a decision tree learning algorithm involves the generation of a series of rules, split points or nodes ordered in a tree-like structure enabling the prediction of a target output value based on the values of the input features. More specifically, XGBoost, as an ensemble technique, is trained by sequentially adding a high number of individually weak but complementary classifiers to produce a robust estimator: each new model is built to be maximally correlated with the negative gradient of the loss function associated with the model assembly [52]. To evaluate the performance of this method we benchmarked it against more standard ensemble learning algorithms such as AdaBoost, Logistic Regression, SVM, and Random Forest.

For each socioeconomic dataset, we trained our models by using 75% of the available data for training and the remaining 25% for testing. During the training phase, the training data undergoes a k-fold inner cross-validation, with $k = 5$, where all splits are computed in a stratified manner

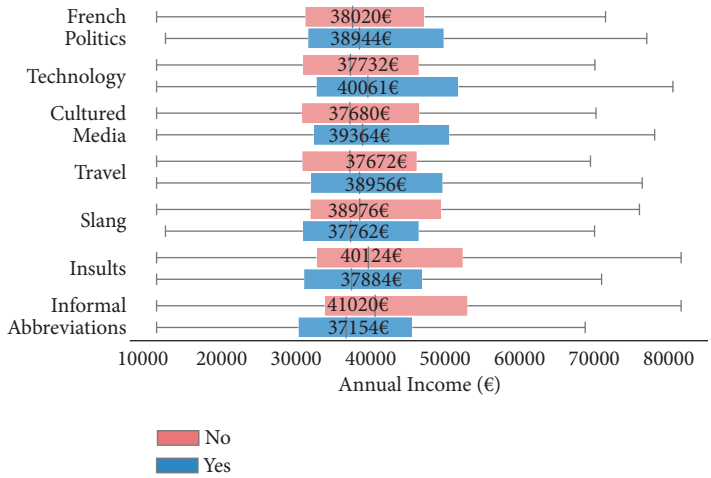


Figure 8: Average income for users who tweeted about a given topic (blue) vs. those who did not (red). Label of the considered topic is on the left.

Table 2: Classification performance (5-CV): AUC scores (mean ± STD) of five different classifiers on each dataset.

	Census	Occupation	Expert
AdaBoost	0.549 ± 0.009	0.628 ± 0.022	0.575 ± 0.013
Logistic Reg.	0.658 ± 0.013	0.778 ± 0.058	0.571 ± 0.033
SVM	0.657 ± 0.016	0.788 ± 0.041	0.600 ± 0.012
Random Forest	0.677 ± 0.011	0.783 ± 0.017	0.593 ± 0.049
XGBoost	**0.700 ± 0.011**	**0.798 ± 0.015**	**0.605 ± 0.029**

to get the same ratio of lower to higher SES users. The four first blocks were used for inner training and the remainder for inner testing. This was repeated ten times for each model so that in the end each model's performance on the validation set was averaged over 50 samples. For each model, the parameters were fine-tuned by training 500 different models over the aforementioned splits. The selected one was that which gave the best performance on average, which was then applied to the held-out test set. This is then repeated through a 5-fold outer cross-validation.

In terms of prediction score, we followed a standard procedure in the literature [53] and evaluated the learned models by considering the area under the receiver operating characteristic curve (AUC). This metric can be thought as the probability that a classifier ranks a randomly chosen positive instance higher than a randomly chosen negative one [52]. This procedure was applied to each of our datasets. The obtained results are shown in Figure 9 and in Table 3.

As a result, we first observed that XGBoost consistently provided top prediction scores when compared to AdaBoost and Random Forest (all performance scores are summarized in Table 2). We hence used it for our predictions in the remainder of this study. We found that the LinkedIn data was the best, with $AUC = 0.80$, to train a model to predict SES of people based on their semantic features. It provided a 10% increase in performance as compared to the census based inference with $AUC = 0.70$, and 19% relative to expert annotated data with $AUC = 0.61$. Thus we can conclude that there seems to be a trade-off between scalability and prediction quality, as while the occupation dataset provided the best results, it seems unlikely to be subject to any upscaling due to the high cost of obtaining a

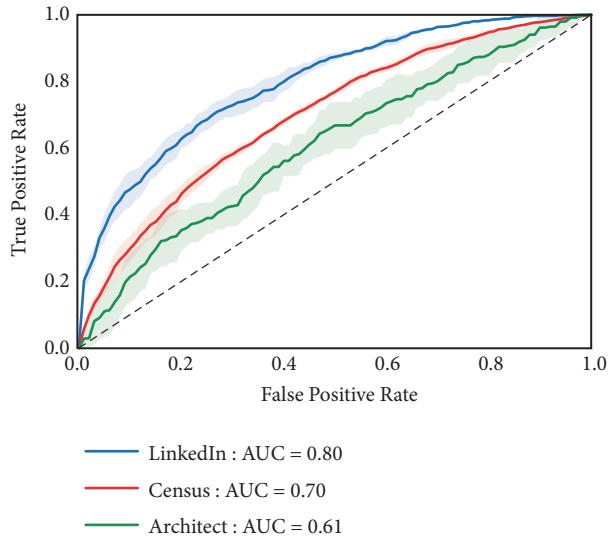


Figure 9: ROC curves for 2-way SES prediction using tuned XGBoost in each of the 3 SES datasets. AUC values are reported in the legend. The dashed line corresponds to the line of no discrimination. Solid lines assign average values over all folds while shaded regions represent standard deviation.

Table 3: Detailed average performance (5-CV) on test data for the binary SES inference problem for each of the 3 datasets.

Dataset	SES Class	Performance on test set		
		Precision	Recall	F1-score
Census	Low	0.652	0.596	0.624
	High	0.628	0.682	0.652
LinkedIn	Low	0.700	0.733	0.717
	High	0.735	0.702	0.720
Architect	Low	0.622	0.598	0.607
	High	0.550	0.573	0.556

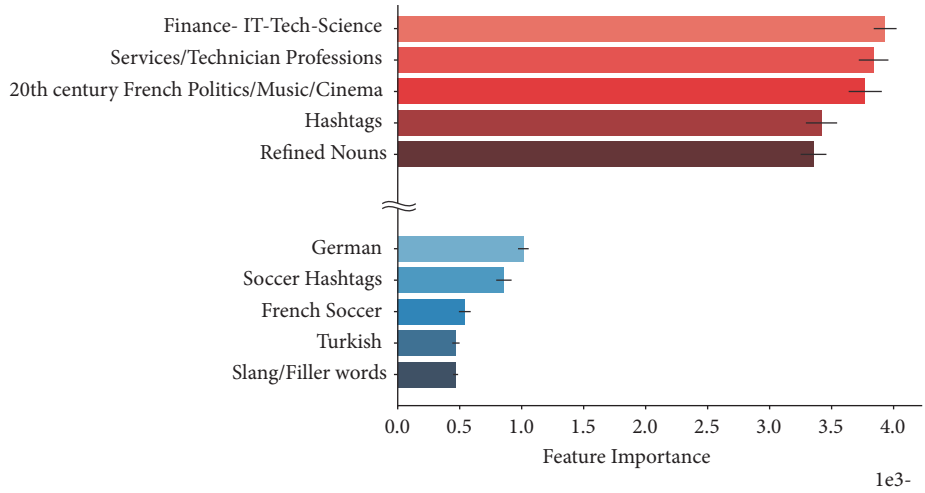


Figure 10: Top (red) and bottom (blue) five topics ranked in terms of their predictive performance in the XGBoost model trained on LinkedIn. Error bars indicate s.d. across the 10 cross-validation samples.

clean dataset. Relying on location to estimate SES seems to be more likely to benefit from such an approach, though at the cost of an increased number of mislabeled users in the dataset. Moreover, the annotator's estimation of SES using Street View at each home location seems to be hindered by the large variability of urban features. Note that even though interagreement is 76%, the Cohen's kappa score for annotator interagreement is low at 0.169. Furthermore, we remark that the expert annotated pipeline was also subject to noise affecting the home location estimations, which potentially contributed to the lowest predictive performance.

We also report the top and bottom five topics ranked by their importance when the XGBoost model was trained on the best performing proxy, i.e., on occupation (see Figure 10). Perhaps unsurprisingly, topics related to professional occupations are the ones recognized by the model as most important. Nevertheless, syntax remains an important feature too. Furthermore, topics associated with particular communities (German/Turkish) or general interest (Soccer) seem to be less useful in terms of SES discrimination. This could in turn be explained by the sparsity of individuals using them or inversely, by the breadth of users discussing them.

Finally, it should also be noted that following recent work by Aletras and Chamberlain in [26], we tested our model by extending the feature set with the *node2vec* embedding of users computed from the mutual mention graph of Twitter. Nevertheless, in our setting, it did not significantly increase the overall predictive performance of the inference pipeline. We hence did not include it in the feature set for the sake of simplicity.

6. Limitations

In this work we combined multiple datasets collected from various sources. Each of them came with some bias due to the data collection and posttreatment methods or the incomplete set of users. These biases may limit the success of our inference; thus, their identification is important for the interpretation and future developments of our framework.

(i) Location Data. Although we designed very strict conditions for the precise inference of home locations of geolocated users, this process may have some uncertainty due to outlier behavior. Further bias may be induced by the relatively long time passed between the posting of the location data and of the tweets collection of users.

(ii) Census Data. As we already mentioned the census data does not cover the entire French territory as it reports only cells with close to 2,000 inhabitants. This may introduce biases in two ways: by limiting the number of people in our sample living in rural areas and by associating income

with large variation to each cell. While the former limit had marginal effects on our predictions, as Twitter users mostly live in urban areas, we addressed the latter effect by associating the median income to users located in a given cell.

(iii) Occupation Data. LinkedIn as a professional online social network is predominantly used by people from IT, business, management, marketing, or other expert areas, typically associated with higher education levels and higher salaries. Moreover, we could observe only users who shared their professional profiles on Twitter, which may further biased our training set. In terms of occupational-salary classification, the data in [40] was collected in 2010 thus may not contain more recent professions. These biases may induce limits in the representativeness of our training data and thus in the predictions' precision. However, results based on this method of SES annotation performed best in our measurements, indicating that professions are among the most predictive features of SES, as has been reported in [11].

(iv) Annotated Home Locations. The remote sensing annotation was done by experts and their evaluation was based on visual inspection and biased by some unavoidable subjectivity. Although their annotations were cross-referenced and found to be consistent, they still contained biases, like overrepresentative middle classes, which somewhat undermined the prediction task based on this dataset.

(v) Different Sets of Users. Our methodologies rely on nonentirely overlapping user sets when they turn to SES inference using occupational data, census data, or remotely sensed values as proxy for individual socioeconomic status. The obtained results are undoubtedly linked to the set of individuals used in each dataset, which may affect the discriminative analysis on the advantages that each proxy provides for the inference task. On the other hand, due to the same collection filters and preprocessing conditions, users in these subsets may be considered similar enough to be able to compare the performance provided by the different methods.

Despite these shortcomings, using all three datasets, we were able to infer SES with performances close to earlier reported results, which were based on more thoroughly annotated datasets. Our results and our approach of using open, crawlable, or remotely sensed data highlight the potential of the proposed methodologies.

7. Conclusions

In this work we proposed a novel methodology for the inference of the SES of Twitter users. We built our models combining information obtained from numerous sources, including Twitter, census data, LinkedIn, and Google Maps. We developed precise methods of home location inference from geolocation, novel annotation of remotely sensed images of living environments, and effective combination of datasets collected from multiple sources. In terms of novelty, we demonstrated that, within the French Twitter space, the utilization of words in different topic categories, identified via advanced semantic analysis of tweets, can discriminate between people of different income and that the mobility patterns and such predictability of users' whereabouts are strongly dependent on SES of people. Furthermore, we showed that among the candidate socioeconomic proxies chosen, the best results were obtained using occupational data. More importantly, we presented a proof-of-concept that our methods are competitive in terms of SES inference when compared to other methods relying on domain specific information. We can identify several future directions and applications of our work. First, further development of data annotation of remotely sensed information is a promising direction. Note that after training, our model requires as input only information that can be collected exclusively from Twitter, without relying on other data sources. This holds a large potential in terms of SES inference of larger sets of Twitter users, which in turn opens the door for studies to address population level correlations of SES with language, space, time, or social network. As such, our methodology has the merit not only of addressing open scientific questions, but also of contributing to the development of new applications in recommendation systems, in predicting customer behavior, or in online social services.

Data Availability

In order to uphold the strict privacy laws in France as well as the agreement signed with our data provider GNIP, full disclosure of the original dataset is not possible. The GitHub repository containing the data collection and preprocessing pipelines is available at https://github.com/jaklevab/TWITTERSES.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

We thank J-Ph. Magué, J-P. Chevrot, D. Seddah, D. Carnino, and E. De La Clergerie for constructive discussions and for their advice on data management and analysis. We are grateful to J. Altnéder and M. Hunyadi for their contributions as expert architects for data annotation. The manuscript was presented at the 2018 IEEE 18th International Conference on Data Mining, IWSC'18 2nd International Workshop on Social Computing (Singapore, 17th November, 2018). This work was supported by the SoSweet ANR project (ANR-15-CE38-0011), the MOTIf Stic-AmSud project (18-STIC-07), and the ACADEMICS project financed by IDEX LYON.

References

[1] V. Mayer-Schönberger and K. Cukier, *Big Data: A Revolution That Transforms How We Work, Live, and Think*, John Murray, 2012.

[2] D. Lazer, A. Pentland, L. Adamic et al., "Life in the network: the coming age of computational social science," *Science*, vol. 323, no. 5915, p. 721, 2009.

[3] K. D. V. Liere and R. E. Dunlap, "The social bases of environmental concern: A review of hypotheses, explanations and empirical evidence," *Public Opinion Quarterly*, vol. 44, no. 2, pp. 181–197, 1980.

[4] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: homophily in social networks," *Annual Review of Sociology*, vol. 27, no. 1, pp. 415–444, 2001.

[5] Y. Leo, E. Fleury, J. I. Alvarez-Hamelin, C. Sarraute, and M. Karsai, "Socioeconomic correlations and stratification in social-communication networks," *Journal of the Royal Society Interface*, vol. 13, no. 125, Article ID 20160598, 2016.

[6] J. L. Brown-Iannuzzi, K. B. Lundberg, and S. McKee, "The politics of socioeconomic status: how socioeconomic status may influence political attitudes and engagement," *Current Opinion in Psychology*, vol. 18, pp. 11–14, 2017.

[7] J. L. Abitbol, M. Karsai, J. Magué, J. Chevrot, and E. Fleury, "Socioeconomic dependencies of linguistic patterns in twitter: a multivariate analysis," in *Proceedings of the World Wide Web Conference (TheWebConf '18)*, pp. 1125–1134, Lyon, France, April 2018.

[8] M. Kosinski, D. Stillwell, and T. Graepel, "Private traits and attributes are predictable from digital records of human behavior," *Proceedings of the National Acadamy of Sciences of the United States of America*, vol. 110, no. 15, pp. 5802–5805, 2013.

[9] Y. Leo, M. Karsai, C. Sarraute, and E. Fleury, "Correlations and dynamics of consumption patterns in social-economic networks," *Social Network Analysis and Mining*, vol. 8, no. 1, p. 9, 2018.

[10] T. Piketty, "Capital in the 21st century," 2014.

[11] D. Preoţiuc-Pietro, S. Volkova, V. Lampos, Y. Bachrach, N. Aletras, and L. A. Braunstein, "Studying user income through language, behaviour and affect in social media," *PLoS ONE*, vol. 10, no. 9, Article ID e0138717, pp. 1–17, 2015.

[12] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010.

[13] J. Levy Abitbol, M. Karsai, and E. Fleury, "Location, occupation, and semantics based socioeconomic status inference on twitter," in *Proceedings of the 18th International Conference on Data Mining (IWSC '18) and 2nd International Workshop on Social Computing (ICDMW '18)*, pp. 1192–1199, November 2018.

[14] J. L. Abitbol, https://github.com/jaklevab/TWITTERSES, 2019.

[15] B. P. Chamberlain, C. Humby, and M. Deisenroth, Detecting the age of twitter users, 2016, https://arxiv.org/abs/1601.04621.

[16] T. Hu, H. Xiao, J. Luo, and T. T. Nguyen, "What the language you tweet says about your occupation," *Tenth International AAAI Conference on Web and Social Media*, 2017.

[17] V. Lampos, N. Aletras, J. K. Geyti, B. Zou, and I. J. Cox, "Inferring the socioeconomic status of social media users based on behaviour and language," in *Advances in Information Retrieval*, Lecture Notes in Computer Science, pp. 689–695, Springer International Publishing, 2016.

[18] D. Preoţiuc-Pietro, V. Lampos, and N. Aletras, "An analysis of the user occupational class through Twitter content," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, pp. 1754–1764, Beijing, China, July 2015.

[19] S. Volkova, G. Coppersmith, and B. Van Durme, "Inferring user political preferences from streaming communications," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL '14)*, pp. 186–196, June 2014.

[20] H. A. Schwartz, J. C. Eichstaedt, M. L. Kern et al., "Personality, gender, and age in the language of social media: the open-vocabulary approach," *PLoS ONE*, vol. 8, no. 9, Article ID e73791, pp. 1–16, 2013.

[21] S. Luo, F. Morone, C. Sarraute, M. Travizano, and H. A. Makse, "Inferring personal economic status from social network location," *Nature Communications*, vol. 8, Article ID 15227, 2017.

[22] Twitter Open API, 2018, https://developer.twitter.com/en/docs.html.

[23] A. Culotta, N. K. Ravi, and J. Cutler, "Predicting the demographics of Twitter users from website traffic data," in *Proceedings of the AAAI Conference on Artificial Intelligence*, January 2015.

[24] J. Eisenstein, B. O'Connor, N. A. Smith, E. P. Xing, and R. C. Berwick, "Diffusion of lexical change in social media," *PLoS ONE*, vol. 9, no. 11, Article ID e113114, pp. 1–13, 2014.

[25] A. Llorente, M. Garcia-Herranz, M. Cebrian, E. Moro, and Y. Moreno, "Social media fingerprints of unemployment," *PLoS ONE*, vol. 10, no. 5, pp. 1–13, 2015.

[26] N. Aletras and B. P. Chamberlain, "Predicting twitter user socioeconomic attributes with network and language information," *Proceedings of the 29th on Hypertext and Social Media*, 2018.

[27] M. C. González, C. A. Hidalgo, and A.-L. Barabási, "Understanding individual human mobility patterns," *Nature*, vol. 453, pp. 779–782, 2008.

[28] R. Jurdak, K. Zhao, J. Liu, M. AbouJaoude, M. Cameron, and D. Newth, "Understanding human mobility from Twitter," *PLoS ONE*, vol. 10, no. 7, 2015.

[29] B. Bernstein, "Language and social class," *The British Journal of Sociology*, vol. 11, no. 3, pp. 271–276, 1960.

[30] R. Compton, D. Jurgens, and D. Allen, "Geotagging one hundred million Twitter accounts with total variation minimization," *IEEE International Conference on Big Data*, 2014.

[31] T. Hu, J. Luo, H. Kautz, and A. Sadilek, "Home location inference from sparse and noisy data: models and applications," *Frontiers of Information Technology & Electronic Engineering*, vol. 17, no. 5, pp. 389–402, 2016.

[32] Gini Index World Bank, 2010, https://data.worldbank.org/indicator/SI.POV.GINI?locations=FR.

[33] INSEE, Revenus, pauvreté et niveau de vie en 2014, 2017, https://www.insee.fr/fr/statistiques/3288151/.

[34] V. Pareto, "Manual of political economy," 1971.

[35] P. Saunders, "Social class and stratification," *Routledge*, 2006.

[36] Y. Xu, A. Belyi, I. Bojic, and C. Ratti, "Human mobility and socioeconomic status: Analysis of Singapore and Boston," *Computers, Environment and Urban Systems*, vol. 72, pp. 51–67, 2018.

[37] LinkedIn, 2018.

[38] LinkedInHelper, 2016, https://linkedhelper.com/.

[39] P. Manzanares-Lopez, J. P. Muñoz-Gea, and J. Malgosa-Sanahuja, "Analysis of linkedin privacy settings: are they sufficient, insufficient or just unknown?" in *Proceedings of the 10th International Conference on Web Information Systems and Technologies (WEBIST '14)*, vol. 1, pp. 285–293, April 2014.

[40] INSEE, "Les salaires dans le secteur privé et les entreprises publiques," 2010, https://www.insee.fr/fr/statistiques/2122237/.

[41] Sequence Matcher Python Library, 2017.

[42] T. Gebru, J. Krause, Y. Wang et al., "Using deep learning and Google Street View to estimate the demographic makeup of neighborhoods across the United States," *Proceedings of the National Acadamy of Sciences of the United States of America*, vol. 114, no. 50, pp. 13108–13113, 2017.

[43] N. Jean, M. Burke, M. Xie, W. M. Davis, D. B. Lobell, and S. Ermon, "Combining satellite imagery and machine learning to predict poverty," *Science*, vol. 353, no. 6301, pp. 790–794, 2016.

[44] Google Maps Static API, 2018, https://developers.google.com/maps/.

[45] M. Castelluccio, G. Poggi, C. Sansone, and L. Verdoliva, Land use classification in remote sensing images by convolutional neural networks, 2015, https://arxiv.org/abs/1508.00092.

[46] UC Merced Land Use Dataset, 2017, http://weegee.vision.ucmerced.edu/datasets/landuse.html.

[47] F. Chollet et al. Keras. https://keras.io, 2015, date of access: November 2018.

[48] J. Deng, W. Dong, and R. Socher, "ImageNet: a large-scale hierarchical image database," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '09)*, pp. 248–255, Miami, FL, USA, June 2009.

[49] J. Leskovec, A. Rajaraman, and J. Ullman, *Mining of Massive Datasets*, Cambridge University Press, 2014.

[50] T. Mikolov, K. Chen, G. Corrado, and J. Dean, Efficient estimation of word representations in vector space, 2013https://arxiv.org/abs/1301.3781.

[51] T. Chen and C. Guestrin, "XGBoost: a scalable tree boosting system," *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[52] L. Torlay, M. Perrone-Bertolotti, E. Thomas, and M. Baciu, "Machine learning–XGBoost analysis of language networks to classify patients with epilepsy," *Brain Informatics*, vol. 4, no. 3, pp. 159–169, 2017.

[53] J. Foster, T. Provost, and R. Kohavi, "The case against accuracy estimation for comparing induction algorithms," in *Proceedings of the 15th International Conference on Machine Learning (ICML '98)*, pp. 445–453, San Francisco, Calif, USA, 1998.

Hindawi
Complexity
Volume 2019, Article ID 4047894, 15 pages
https://doi.org/10.1155/2019/4047894

Research Article

Semantic-Aware Top-k Multirequest Optimal Route

Shuang Wang, Yingchun Xu, Yinzhe Wang, Hezhi Liu, Qiaoqiao Zhang, Tiemin Ma, Shengnan Liu, Siyuan Zhang, and Anliang Li

Software College, Northeastern University, Shenyang 110004, China

Correspondence should be addressed to Shuang Wang; wangsh@mail.neu.edu.cn

Received 25 January 2019; Accepted 3 April 2019; Published 15 May 2019

Guest Editor: Jianxin Li

In recent years, research on location-based services has received a lot of interest, in both industry and academic aspects, due to a wide range of potential applications. Among them, one of the active topic areas is the route planning on a point-of-interest (POI) network. We study the top-k optimal routes querying on large, general graphs where the edge weights may not satisfy the triangle inequality. The query strives to find the top-k optimal routes from a given source, which must visit a number of vertices with all the services that the user needs. Existing POI query methods mainly focus on the textual similarities and ignore the semantic understanding of keywords in spatial objects and queries. To address this problem, this paper studies the semantic similarity of POI keyword searching in the route. Another problem is that most of the previous studies consider that a POI belongs to a category, and they do not consider that a POI may provide various kinds of services even in the same category. So, we propose a novel top-k optimal route planning algorithm based on semantic perception (KOR-SP). In KOR-SP, we define a dominance relationship between two partially explored routes which leads to a smaller searching space and consider the semantic similarity of keywords and the number of single POI's services. We use an efficient label indexing technique for the shortest path queries to further improve efficiency. Finally, we perform an extensive experimental evaluation on multiple real-world graphs to demonstrate that the proposed methods deliver excellent performance.

1. Introduction

In recent years, the rapid advancements of wireless communication techniques, Global Positioning System (GPS), and smart mobile devices have enabled a lot of Location-based Services (LBS). Among them, one of the popular issues is the path/route planning in a point-of-interest (POI) network [1, 2]. The users of the LBS often want to find short routes that pass through multiple POIs; consequently, developing trip planning queries that can find the shortest routes that passed through user-specified categories has attracted considerable attention [3, 4]. While the problem of computing the optimal route has been extensively studied and many efficient techniques have been developed over the past several decades, most of the past studies on route planning focused on origin-destination route planning and did not consider the user's specific requirements.

Recently, there are some approaches that find the route by using the user's queries. However, the approaches may find a longer route than the one that meets the user's actual demands, because the query keywords only show the meaning of user's query rather than requiring the conformance in shape. A major problem with the existing approaches is that they only output routes that perfectly match the given categories [5–8]. Take Figure 1 as an example: each object can be viewed as a POI that has a spatial location and additional keywords. Considering a user who wants to watch a film and she issues a keyword query q with her current location and keyword *film*, if we apply the traditional spatial keyword query method, o_1 is returned as it contains the query keyword. However, we can find that o_6 also meets the user's requirement as we know that the user only wants to watch a film. To overcome this problem, we introduce flexible semantic matching based on POI keywords to find shorter routes in a flexible manner. In addition, each POI contains a lot of keywords and provides multiple services. A POI may cover multiple keywords in query; otherwise, each POI only corresponds to one of the keywords in query. For example, a POI, called *WanDa Plaza*, has a lot of keywords as *cinema, popcorn, food*, etc. If a user is looking for POIs where she

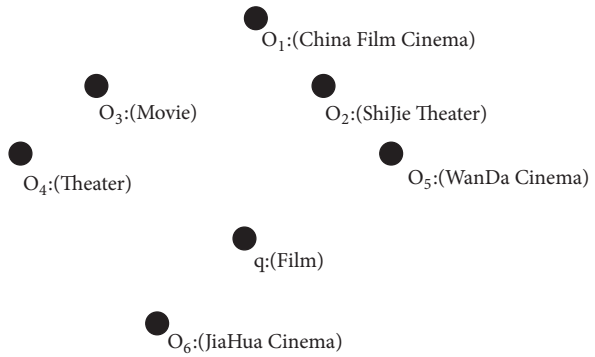


Figure 1: An example of keyword query.

can eat something and see a movie, she issues a query q with keywords *movie* and *food*. We can know that this POI may meet the all requirements of the user; otherwise, she must visit a restaurant and a cinema, respectively. So, we can cut the length of route down by reducing the number of POIs in the route.

Existing approaches find the shortest route that is an optimal sequenced route, but these approaches result in a lack of flexibility in route planning and leave user without possibility of choice. We are proposing to the top-k algorithm in order to provide more choices and satisfy users to the maximum. Besides, compared with keyword ordered query, keyword unordered query is more flexible, and we only need to consider the distance between POI and the current point under the premise of meeting user requirements. At the same time, unordered query can avoid the distant POI becoming the nearest neighbor because the query order of the keywords is no longer considered during the query process.

In this paper, solving the top-k route search problem faces three challenges. The first challenge is the larger search space of the query. Because we consider the semantic relation of the queries and POIs, while not the string matching, then the number of candidate objects is larger than the existing approaches. It calls for effective methods to filter some candidates for avoiding exhaustive search. The second challenge is the strategies to extend the route. Many existing methods only consider the nearest neighbor of the current point, but the route generated by extending from their neighbor perhaps does not become the final optimal result. In this paper, we not only consider the distance between the current object and the neighbors, but also need to take the neighbor as the current object and consider its neighbor for which the distance is the smallest. It shows that the extending route by this method has higher probability to be the final optimal route. Here, we consider the semantic distance and spatial distance simultaneously. In order to efficiently compute the distance cost, we propose a method to use the 2-hop labeling technique [9–12]. The third challenge is route refinement mechanism. The POIs in the final route found by our method may be redundant; that is to say, perhaps more than two POIs provide the same services in one route, since our algorithm is greedy approach. So, we need to propose a refinement mechanism to further enhance the route quality.

The main contributions of this paper can be summarized as follows:

(1) We introduce a semantic similarity to the route search query, which allows us to search for routes flexibly

(2) We propose the *top-k optimal route based on semantic perception* (KOR-SP), which finds all preferred routes related to keyword with semantic perception

(3) We propose a method to find the x-th nearest neighbor based on semantic perception

(4) We use real-world POI datasets to test and prove the superiority of the algorithm.

The remainder of this paper is organized as follows. In Section 2, we briefly review the related work. In Section 3, we formally state the problem. In Section 4, we first introduce the KOR-SP algorithm and how to find the x-th nearest neighbor. The empirical performance study is presented in Section 5. Conclusion and future work are presented in Section 6.

2. Related Work

We review the related works in this section. Route planning is one of the hot topics on LBSs [13, 14]. The algorithms on *destination-oriented route planning* have been split into *single-destination route planning* and *multidestination route planning*. Among them, a number of algorithms belonging to *single-destination route planning*, such as *Dijkstra* [15] and A^* [16], have been proposed to find the shortest route between two locations. Besides, an increasing number of approaches on *multidestination route planning* have been proposed [17–19], such as *Traveling Salesman Problem* (TSP) [19] and *TSP with Neighborhood* (TSPN) [17]. All of the above are *destination-oriented route planning,* but *requirement-oriented route planning* is another kind of routing problem. Li [18] et al. proposed *Trip Planning Query* (TPQ) and proposed *Nearest Neighbor Algorithm* (A_{NN}) and *Minimum Distance Algorithm* (A_{MD}). A_{NN} visits the nearest POI that belongs to the last visited POI and A_{MD} finds each "good" POI that belongs to each unvisited category and traverses these POIs in a nearest neighbor order. However, the planned results of these algorithms are not good since the routes found by these algorithms may be tortuous which means that they are full of twists, turns, or bends. Based on TPQ, Ahmadi and Nascimento [20] studied *Sequenced Group Trip Planning Queries* (SGTPQs) to find a sequence of POIs belonging to the specified categories and minimize the total distance travelled by all groups of users. Sharifzadeh [7] et al. proposed a related query problem named *Optimal Sequenced Route* (OSR) to retrieve the shortest route from a given source via several locations with different categories in a particular order. Based on OSR, Liu [21] et al. proposed top-k optimal sequenced route (KOSR) to find the top-k optimal routes from a given source to a given destination, which must visit a number of POIs with specific categories in a particular order. However, the above works considered that a POI only belongs to a category. In an urban area, a POI may not only belong to a category but also provide various services. A better routing approach should consider whether the provided services on the route satisfy user's requests rather than the categories.

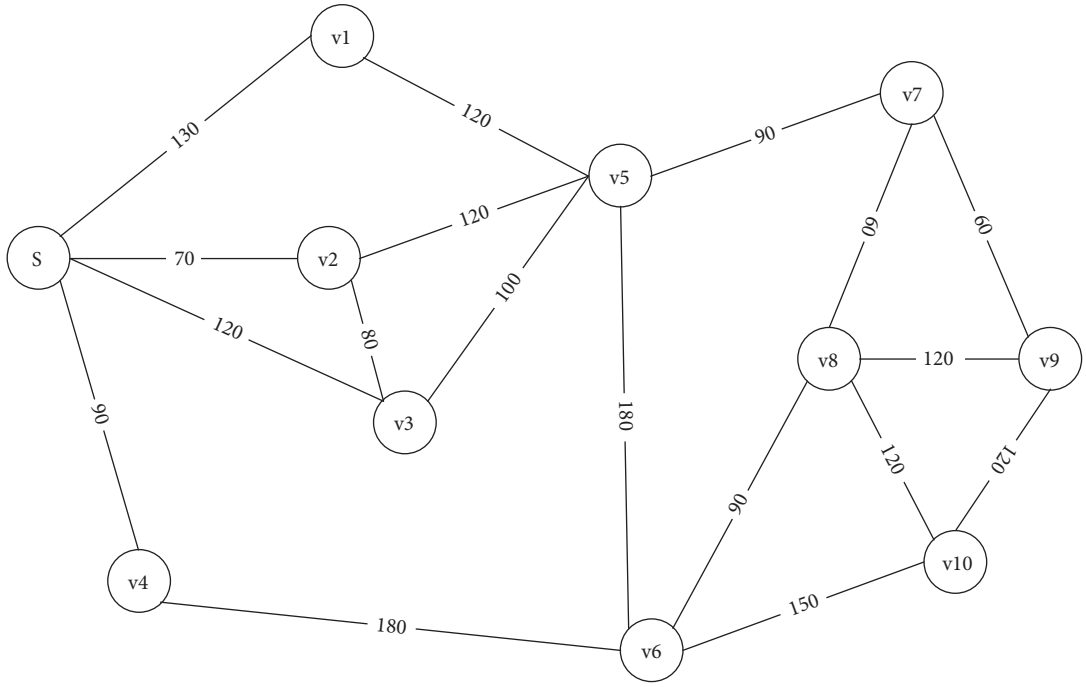


Figure 2: A route network.

In addition, there is also a lot of research on POI. POI recommendation is one of hot topics and it can provide better POIs for route planning. Lei Tang [22] et al. proposed a personal POI recommendation method based on destination prediction. And Jianxin Li [23] proposed *Personalized Influential Topic Search,* or more succinctly PIT-Search. The goal of PIT-Search is to find how important topics and influential users might be better leverage to meet a specific user's information need.

Route planning has attracted a lot of attention. So far, people still focus on user preferences or POI's categories to extend their work. But in reality, the POI's categories are not able to sufficiently represent services provided by POI. So, in order to better meet the needs of the user, it is necessary to consider the services provided by POI when designing the algorithm. So, in this paper, we designed a multirequest route planning algorithm considering the POI's service.

3. Problem Statement

We formalize the KOR-SP problem in this section. Frequent notations are summarized in Table 2.

3.1. Some Definitions. In this section, we first define some terms used in this paper and then specify our research problem.

Definition 1 (graph). An undirected weighted graph $G(V, E, W)$ consists of a set of edge weights that represent the distance between two POIs including a vertex set V and an edge set $E \subseteq V \times V$. Weight function $W : E \longrightarrow R^+$ takes an edge (u, v) as input and returns a nonnegative cost of the edge $W((u, v))$. For example, in Figure 2, we have $W((s, v_4)) = 90$. Note that the edge weights can be arbitrary and may not satisfy the triangle inequality. At the same time, it is also applicable to directed weighted graph and the edge weights can represent distance, time and so on.

Definition 2 (request). Request means a thing, a need, a requirement, or a service that the user wants. $Q = \{q_1, q_2, \ldots, q_{|R|}\}$ denotes the collection of requests.

Definition 3 (POI). POI, the abbreviation of point of interest, represents the specific location in the map. It includes two aspects information: spatial information and key words. It is defined as follows:

$$v = (v.\lambda, v.\kappa), \tag{1}$$

where v is the POI, $v.\lambda$ is the POI's spatial information, usually in the form of latitude and longitude of POI, and $v.\kappa$ is the set of keywords.

It is worth mentioning that each POI may contain multiple keywords, and through these keywords it can get the basic information of the POI. For example, there is a POI named *WanDa* that contains keywords as *movie, food,* and so on. These keywords can describe the basic characteristics of the POI. And these keywords of POI can be defined as follows:

$$v.\kappa = \left\{\kappa_1, \kappa_2, \ldots, \kappa_i, \ldots, \kappa_{|v.\kappa|}\right\}, \tag{2}$$

where k_i is the i-th keyword of POI and $k_{|v|}$ is the total number of POI's keywords.

In addition, we need to consider the semantics of the keyword when querying the POI. For each keyword, we acquire its topics through the *Latent Dirichlet Allocation* (LDA) [24] and set up a collection to hold these topics and every topic's probability. This can be defined as follows:

$$\kappa = \{T, \eta\}, \tag{3}$$

where T and η are the set of topics and topic's probability, respectively.

It is worth mentioning that we use the probabilistic topic model to transform the textual description into their semantic representations, and then we can use them to quantify the semantic correlation between textual descriptions. By applying a popular probabilistic topic model called LDA, we can obtain a topic distribution of each object to describe the semantic correlation between the object keywords and a limited set of potential topics. Given a query and an object, it is possible to measure their semantic similarity based on their topic distributions.

Definition 4 (keyword similarity). Given a query keyword k and a POI's keyword q, the similarity $sim(k, q) \in [0, 1]$ is calculated by an arbitrary function such as the *Wu and Palmer similarity* or length [7, 15] . We assume the relations in the similarity as follows:

$$sim(k, q) = \begin{cases} \beta & , \beta > 0.5 \\ 0 & , else, \end{cases} \tag{4}$$

where k is the query keyword and q is the POI's keyword. If k is relevant to q and corresponding probability $\beta > 0.5$, we set $sim(k, q) = \beta$; otherwise, $sim(k, q) = 0$. And when we calculate the value β as in the following, k and q are topic probability distribution vectors representing the query keyword and the POI's keyword, respectively:

$$\beta = \cos(k, q) = \frac{k \bullet q}{\|k\| * \|q\|} = \frac{\sum_{i=1}^{n} k_i \times q_i}{\sqrt{\sum_{i=1}^{n} k_i^2} \times \sqrt{\sum_{i=1}^{n} q_i^2}}. \tag{5}$$

For example, each tuple in Table 3 is a topic distribution over five topics. Considering a user who wants to watch a movie and she issues a keyword query q with her current location and keyword *cinema*, we can get that $\beta_{cinema,theater} = \cos(cinema, theater) = 0.995$ and the value is larger than 0.5, so the semantic similarity is $sim(cinema, theater) = 0.995$.

Definition 5 (PRQ and QRP). $PRQ(q) = \{v \mid v \in V\}$ provides the set of POIs that is the services with which it can meet user's querying keyword q and $QRP(v) = \{k \mid k \in v.\kappa\}$ provides the set of point's services. Given a $q \in Q$ and a $v \in V$, we can get that $PRQ(q) \subseteq V$ and $QRP(v) \subseteq Q$. Take Table 1 as an example: $PRQ(q_1) = \{v_1, v_3, v_4\}$ and $QRP(v_1) = \{q_1, q_2\}$.

Definition 6 (route). Route refers to a collection containing several POIs. POI in the collection has a certain order to form a route, so the route is defined as follows:

$$R = \{v_1, v_2, \ldots, v_j, \ldots, v_{|R|}\}, \tag{6}$$

Table 1: POI information.

POI	Provided Services
v_1	q_1, q_2
v_2	q_3, q_9
v_3	q_1, q_4, q_5
v_4	q_1, q_2
v_5	q_6
v_6	q_6, q_7, q_9
v_7	q_3, q_9
v_8	q_2, q_7
v_9	q_5, q_7, q_8
v_{10}	q_3

where v_i is the i-th POI in the route and |R| is the number of POIs in the route.

And each route also has its keywords, because the route contains several POIs, so the route also contains the keywords of all POIs, which can be expressed as follows:

$$R.K = \bigcup_{v \in R} v.\kappa. \tag{7}$$

As you can see, the keyword of the route is a collection of all the POI's keywords in the route.

Definition 7 (route cost). Given a set of user's requests $Q = \{q_1, q_2, \ldots, q_n\}$ and a route $R = \{s, v_1, \ldots, v_m\}$, the cost of a route is the spatial distance through all the POIs in the entire route from a given source s. We can denote the cost of route R as follows:

$$\cos t(R) = dist(s, v_1) + \sum_{i=1}^{|R|-2} dist(v_i, v_{i+1}), \tag{8}$$

where $cost(R)$ is the spatial distance of the route, $dist(s, v_1)$ is the distance between the starting point s and the first POI in route, and $dist(v_i, v_{i+1})$ is the distance of the i-th POI to the $(i + 1)$-th POI in the route.

Definition 8 (route average cost). Because one POI may include multiple services and requests, we should consider the number of services in route when extending a partial route. So, the best method is that, calculating the average cost that is the route cost divided by the number of services, we consider the partial route with the least average cost to extend. Given a route $R = \langle s, v_1, v_2, \ldots, v_i, \ldots, v_{|R|} \rangle$ and a set of user's requests $Q = \{q_1, q_2, \ldots, q_n\}$, we denote the average cost as follows:

$$ave(R) = \frac{\cos t(R)}{\left| \bigcup_{i=1}^{|R|} QRP(v_i) \cap Q \right|}. \tag{9}$$

3.2. Two-Hop Labeling Technique. The POI map data is stored on disk. To answer user queries rapidly with low I/O access and speed-up distant cost computation, we build index HI stored on disk.

Table 2: Meaning of notations.

Notation	Meaning	Notation	Meaning
$R_{s,t}$	A route from s to t	K	Top k results are needed
Q	The requests of user	$D_{s,t}$	The distance from s to t
C	The collection of POIs	$v.\lambda$	POI's spatial information
$\|NR\|$	The number of services that meet user's request in route or witness R	$v.\kappa$	POI's keywords
$\|NQ\|$	The number of user's requests	v	POI point of interest

Table 3: Topic distributions of textual descriptions.

Keyword	Topics				
	Exercise	Movie	Drink	Shop	Food
market	0.09	0.09	0.09	0.64	0.09
fast food	0.04	0.04	0.16	0.04	0.72
cinema	0.07	0.72	0.07	0.07	0.07
Wal-Mart	0.07	0.07	0.07	0.72	0.07
theater	0.04	0.84	0.03	0.04	0.04
KFC	0.03	0.03	0.03	0.03	0.88

Table 4: 2-hop index HI.

Vertex	HI(v)
s	$(s,0),(v_2,70),(v_4,90),(v_3,120),(v_1,130)$
v_1	$(v_1,0),(v_5,120),(s,130)$
v_2	$(v_2,0),(v_3,80),(v_5,120)$
v_3	$(v_3,0),(v_5,100)$
v_4	$(v_4,0),(v_6,180)$
v_5	$(v_5,0),(v_7,90),(v_6,180)$
v_6	$(v_6,0),(v_{10},150)$
v_7	$(v_7,0),(v_8,60),(v_9,60)$
v_8	$(v_8,0),(v_9,120),(v_{10},120)$
v_9	$(v_9,0),(v_{10},120)$
v_{10}	$(v_{10},0)$

Table 4 shows the HI for the POI map in Figure 2; for each vertex $v \in V$, 2-hop labeling maintains a label HI(v). In particular, HI(v) consists of a set of label entries in the form of $(u, d_{u,v})$, where $u \in V$ is a vertex that is able to reach v and $d_{u,v} = dist(u, v)$.

We note that it is NP-hard to construct 2-hop tags at a minimum size while satisfying the coverage feature. Therefore, the existing methods [9–11, 25] are all heuristic and approximate the minimal 2-hop labeling index. Alternatively, we can use the full pair shortest path algorithm to generate the index. Although it works, it requires an index size of $O(|V|^2)$, which is unacceptable for large graphs.

4. Proposed Solutions

In this section, we provide the effective method of solving KOR-SP. We described a route planning method based on semantic perception that satisfies multiple requests of user. The method proposes a dominating relationship on candidate routes to filter the candidate routes and thus reduces the search space. In addition, by combining an optimization technique, we can effectively find the x-th nearest neighbor of the current vertex.

4.1. KOR-SP Algorithm. We introduce the domination relationship: the so-called domination relationship is in the same starting point and end point, and the route with larger average cost dominates the small one, as shown in Figure 2, when the user demands service for $\{q_1, q_3, q_6, q_7, q_8\}$ when considering $\langle s \longrightarrow v_1 \longrightarrow v_5\rangle$ (2,250) and $\langle s \longrightarrow v_3 \longrightarrow v_5\rangle$2,220). In the case of the same destination, the numbers of services of route $\langle s \longrightarrow v_1 \longrightarrow v_5\rangle$ and $\langle s \longrightarrow v_3 \longrightarrow v_5\rangle$ provided are equal, but route $\langle s \longrightarrow v_3 \longrightarrow v_5\rangle$ has smaller distance cost, so the $\langle s \longrightarrow v_3 \longrightarrow v_5\rangle$ belongs to the dominating path, and $\langle s \longrightarrow v_1 \longrightarrow v_5\rangle$ belongs to the dominated path.

Definition 9 (domination). Given a user's request $Q = \{q_1, \ldots, q_j\}$ and two partially explored candidate routes $R_1 = \langle s, v_1^1, \ldots, v_q^1\rangle$ and $R_2 = \langle s, v_1^2, \ldots, v_q^2\rangle$ $(1 \le q \le j)$, if $v_q^1 = v_q^2$ and $\mathrm{QRP}(R_1) \cap Q = \mathrm{QRP}(R_2) \cap Q$ and $\cos t(R_1) \le \cos t(R_2)$ holds, R_1 dominates R_2, denoted as $R_1 \prec_Q R_2$.

Lemma 10. *Given a KOR-SP query $(s, Q = \langle q_1, \ldots, q_j\rangle, k)$ and two partially explored routes R_1 and R_2, if $R_1 \prec_Q R_2$, then $ave(R_1^*) \le ave(R_2^*)$, where R_1^* and R_2^* are the optimal feasible routes that are extended from R_1 and R_2, respectively.*

Proof. Suppose $R_1 = \langle s, v_1^1, \ldots, v_q^1\rangle$, $R_2 = \langle s, v_1^2, \ldots, v_q^2\rangle$, and $R_1^* = \langle s, v_1^1, \ldots, v_q^1, v_{q+1}^1, \ldots, v_j\rangle$; since R_1^* is the optimal feasible route extended from R_1, $R = \langle v_1^q, v_{q+1}, \ldots, v_j\rangle$ must be the optimal route from v_1^q to v_j. Because $R_1 \prec_Q R_2$, we have $v_q^1 = v_q^2$, $\mathrm{cost}(R_1) < \mathrm{cost}(R_2)$ and the services provided by route are the same; thus, R_2^* can be represented by $\langle s, v_1^2, \ldots, v_q^2, v_{q+1}, \ldots, v_j\rangle$, and then $\mathrm{cost}(R_1^*) = \mathrm{cost}(R_1) + \mathrm{cost}(R)$ and $\mathrm{cost}(R_2^*) = \mathrm{cost}(R_2) + \mathrm{cost}(R)$, and since

Input: Graph: G(V,E); Request: Q = $\langle r_1, r_2, \ldots, r_q \rangle$; number of routes: K;
Output: top-k routes
1 $\forall v \in V$, initialize v.$HT_{<Q}$ and v.$HT_{>Q}$;
2 $\Psi \longleftarrow \emptyset$;
3 priority queue R $\longleftarrow$ {($\langle s \rangle$, 1)};
4 while R is not empty and $|\Psi| < K$ do
5 P=($\langle v_0, v_1, \ldots, v_{q-1}, v_q \rangle$,x)$\longleftarrow$R.extractMin();
6 Q' $\longleftarrow \sum_{i=1}^{|q|} QRP(v_i) \cap Q$
7 if |Q'| = |NQ| then
8 $\Psi \longleftarrow \Psi U\{R\}$;
9 for each i=1,. . .,q-1 do
10 if QRP($\langle v_0, \ldots, v_i \rangle$)=QRP($v_i$.$HT_{<Q}$.getValue())
11 then
12 P'=($\langle v_0, v'_1, \ldots, v_i \rangle$,-) $\longleftarrow$ v_i.$HT_{>Q}$.getValue().extractMin();
13 R.insert(P');
14 v_i.$HT_{<Q}$.remove();
15 else
16 if QRP(p)=QRP (v_q.$HT_{<Q}$.Value) then
17 v_q.$HT_{<Q}$.add(|QRP(P)|, $\langle v_0, \ldots, v_q \rangle$, QRP(P));
18 $v_{q+1} \longleftarrow$ FindNN(v_q,QPR(Q-Q'),1);
19 R.insert(($\langle v_0, v_1, \ldots, v_q, v_{q+1} \rangle$,1));
20 else
21 v_q.$HT_{>Q}$.add(|QRP(P)|,P);
22 if q>0 then
23 $v'_q \longleftarrow$ FindNN(v_{q-1},PRQ(R-QRP($\langle v_0, \ldots, v_{q-1} \rangle$)),x+1);
24 R.insert(($\langle v_0, v_1, \ldots, v_{q-1}, v_q \rangle$,x+1));
25 return Ψ;

Algorithm 1: KOR-SP(G,s,Q,k).

cost(R_1) < cost(R_2), we have cost(R_1^*) < cost(R_2^*) and the services are the same. □

According to Lemma 10, before the optimal potential route expanded from their dominating route to be one of the top-k optimal routes, there is no need to extend the routes that are dominated. Based on dominating relationship, we put forward KOR-SP method (Algorithm 1).

To check relationship of domination and store dominated route, for each POI v, we recommend two hash tables in the shape of (*key*, *value*) pairs. The first is $HT_{<Q}$ for saving dominating route, where key is the number of services that meet the user's requests, provided by the partially dominating route that has been extended from current POI v and explored, and *value* is the route itself. Another one is $HT_{>Q}$ for saving dominated routes, where *key* is the number of services which meet the request of user, provided by the partially explored dominated route that has been extended from v, and *value* is the route itself, and the dominated routes are ordered according to their average costs in an ascending order. We also keep Ψ as a result set to save top-k optimal routes and a global priority queue R for partially explored routes sorted by their average costs in an ascending order. In addition, for each $P = \langle v_0, v_1, \ldots, v_{q-1}, v_q \rangle$, we introduce an additional attributes x to represent that v_q is the x-th nearest neighbor of v_{q-1} when generating P. Initially, only the source with $x = 1$ is added to the queue R. Then, we begin a loop until R is empty or top-k optimal sorting route has been found.

Pruning Dominated Routes. At each iteration, the algorithm chooses the route with the minimum average cost to be checked. If it has completed all of the user's requests, we will add it to Ψ and reconsider dominated routes (lines 5-14). Otherwise, we inspect if it is dominated or not. For a route $P = \langle v_0, v_1, \ldots, v_{q-1}, v_q \rangle$ to be examined, if P is the first route with QRP(P) that reaches vertex v_q, we add p to $HT_{<Q}$ of v_q and extend it via v_q's nearest neighbor v_{q+1} (lines 14-17). Otherwise, if its QRP(P) belongs to the $HT_{<Q}$ of v_q, it signifies that existing other route with QRP(P) and smaller average cost has been maintained and expanded to the v_q, so that P is dominated. According to Lemma 10, there is no need to extend P anymore; therefore, we add it into $HT_{>Q}$ of v_q rather than the priority queue R (lines 19). Then, we generate a new candidate route from P. Because the x-th nearest neighbor of v_{q-1} has generated in the previous iteration, we need to find the $(x + 1)$-th nearest neighbor of v_{q-1} by invoking algorithm FindNN and create candidate route $\langle v_0, v_1, \ldots, v_{q-1}, v_q' \rangle$ with incremental x and insert it into the priority queue (lines 22-24).

Table 5: Running example of Algorithm 1 for Figure 1.

Step	Routes(route(|NR|, ave(R)),x)
1	$(\langle s\rangle(0,0),1)$
2	$(\langle s, v_2\rangle(1,70),1)$
3	$(\langle s, v_2, v_3\rangle(2,75),1),(\langle s, v_4\rangle(1,90),2)$
4	$(\langle s, v_2, v_3, v_5\rangle(3,83.3),1),(\langle s, v_4\rangle(1,90),2),(\langle s, v_2, v_5\rangle(2,95),2)$
5	$(\langle s, v_2, v_3, v_5, v_9\rangle(5,80),1),(\langle s, v_4\rangle(1,90),2),(\langle s, v_2, v_5\rangle(2,95),2),(\langle s, v_2, v_3, v_6\rangle(4,107.5),2)$
6	$(\langle s,v_2,v_3,v_5,v_9\rangle(5,80),1),(\langle s, v_4\rangle(1,90),2),(\langle s, v_2, v_5\rangle(2,95),2),(\langle s, v_2, v_3, v_6\rangle(4,107.5),2)$
7	$(\langle s, v_4\rangle(1,90),2),(\langle s, v_2, v_5\rangle(2,95),2),(\langle s, v_2, v_3, v_6\rangle(4,107.5),2)$
8	$(\langle s, v_4, v_6\rangle(3,90),1),(\langle s, v_2, v_5\rangle(2,95),2),(\langle s, v_2, v_3, v_6\rangle(4,107.5),2),(\langle s, v_3\rangle(1,120),3)$
9	$(\langle s, v_2, v_5\rangle(2,95),2),(\langle s, v_4, v_6, v_{10}\rangle(4,105),1),(\langle s, v_2, v_3, v_6\rangle(4,107.5),2),(\langle s, v_3\rangle(1,120),3),(\langle s, v_4, v_{10}\rangle(2,210),2)$
10	$(\langle s, v_2, v_5, v_9\rangle(4,85),1),(\langle s, v_4, v_6, v_{10}\rangle(4,105),1),(\langle s, v_2, v_3, v_6\rangle(4,107.5),2),(\langle s, v_3\rangle(1,120),3),(\langle s, v_4, v_{10}\rangle(2,210),2)$

Rethink Dominated Route. After finding the optimal route P, we need to rethink the partial routes that has been explored and been dominated by subroutes of P, because these routes are more likely to be extended to be another optimal route now. Therefore, for every POI v_i in P, if $\langle v_0, \ldots v_i\rangle$ dominates the routes with QRP(p_i) in the $HT_{\succ Q}$ of v_i (line 10), we only consider the dominated route p' with the least average cost and at the end of v_i, because other routes at the end of v_i are dominated by P'. This also accounts for why we use a priority queue as *value* in hash table $HT_{\succ Q}$. Since p's $x + 1$ nearest neighbor has been computed after it is dominated, we set its x to "-" (which means it makes no sense generating candidate route) and add it to the priority queue (lines 10-13). Meanwhile, we remove $\langle v_0, \ldots v_i\rangle$ from the $HT_{\prec Q}$ of v_i; thus, the next candidate route can be extended (line 14).

Example 11 (consider Figure 2). Suppose the given query is $(s, \langle r_1, r_3, r_6, r_7, r_8\rangle, 2)$. Table 5 shows the routes in the priority queue R at each step. At step 1, route $\langle s\rangle$ is added to the queue, and then it is extended via v_2 (s's nearest neighbor in C) that is the collection of POIs with the unfinished services in Q, and no candidate route can be generated. At step 2, $\langle s, v_2\rangle$ is examined, it is extended via v_3 (v_2's nearest neighbor in C) that is the collection of POIs with the unfinished services and candidate route $\langle s, v_4\rangle$ is generated via s's 2^{nd} nearest neighbor in C that is the collection of POIs with the unfinished services. And so on until the exit condition is met.

Finding the x-th Nearest Neighbor. Next, we interpreted how to find the x-th nearest neighbor, the core operation in KOR-SP.

Definition 12 (neighbor distance). Because the POI keywords and the query keywords have a semantic difference, we are not able to choose the nearest neighbor according to the actual distance. And then, we calculate the neighbor distance by combining the semantic difference with the actual distance. And we choose the nearest neighbor according to the neighbor distance. We measure the neighbor distance by

$$sem\left(v, v'\right) = \sum_{1}^{m} \left\{\left(1 - sim\left(k, q\right)\right) \bullet dist\left(v, v'\right)\right\}, \tag{10}$$

where v is the current vertex, v' is the possible nearest neighbor, and m is the number of $sim(k, q)$ that is not equal to 0. Besides, k is keyword of v' and q is the query keyword. Now, given a route $R = \langle s, v_1, \ldots, v_i\rangle$, when we extend R to v_j, we can estimate the cost of v_j as follows:

$$\begin{aligned} T_{i,j} = {} & dist\left(v_i, v_j\right) + sem\left(v_i, v_j\right) \\ & + \min\left\{d \mid d \in \left(HI\left(v_j\right) - \left(v_i, d_{v_i, v_j}\right)\right)\right\}, \end{aligned} \tag{11}$$

where v_i is the current node and v_j represents one of the neighbors of v_i. For example, in Figure 2, we can know that $dist(v_2, v_3) = 80$, $sem(v_2, v_3) = 0$, and $\min(d) = 100$, so $T_{2,3} = 180$.

A straightforward way to find the x-th nearest neighbor of vertex v_i in collection of POIs with services in Q-$Q \cap QRP(\langle v_0, \ldots, v_i\rangle)$ is by using 2-hop labeling technique rather than *Dijkstra*'s search, since FindNN is frequently invoked. Frequent Dijkstra searches on large graphs are practically inefficient. When the number of unfinished services is greater than 1, we do the following steps (lines 3-7). We start from v_i and extend vertices via the equation (line 6) that calculates the average distance between two points, and each vertex's average distance with the current vertex is stored in the ascending sorting queue N (line 7). When the number of unfinished services is less than 1, we perform the following steps (lines 9-11). We directly consider the actual distance between the current point and the nearest possible neighbor as the average distance (line 10) and store it in the queue N (line 11). Finally, we output the corresponding vertex as needed (lines 13-14).

Input: Vertex v_i, collection of POIs with service in Q- $Q \cap QRP(\langle v_0, \ldots, v_i \rangle)$ C, integer x.
Output: The x-th nearest neighbor of v_i in C.

```
1  N ⟵ Ø
2  C ⟵ PRQ(Q- Q ∩ QRP(⟨v_0, ..., v_i⟩));
3  if |Q- Q ∩ QRP(⟨v_0, ..., v_i⟩)| > 1 then
4    for v' in C
5      v_j = min(HI(v').d_{u,v'}).get(u)
6      average = T_{v_i,v'} / |QRP(v') ∪ QRP(v_i) ∩ C|;
7      N.add(v',average);
8  else
9    for v' in C
10     average = dist(v_i, v')
11     N.add(v',average);
12   N.ascending(average);
13   if |N| ≥ x then
14     return N[x];
```

Algorithm 2: FindNN(v_i,C,x).

4.2. Approximate KOR-SP

4.2.1. Domination Conditions. This is different from KOR-SP. The dominating relationship changes such that it does not require the partial routes that have been queried to provide the same services. Next, we will introduce the novel dominating relationship in detail.

We reconsider the dominating relationship. The original requirements are too strict. First, some partial explored routes should have the same end; second, they should provide the same number of the services required by user. For example, in Figure 2, $\langle s \longrightarrow v_1 \longrightarrow v_5 \rangle$ (2,250) and $\langle s \longrightarrow v_2 \longrightarrow v_3 \longrightarrow v_5 \rangle$(3,250) have the same destination, and the numbers of services of route $\langle s \longrightarrow v_1 \longrightarrow v_5 \rangle$ and $\langle s \longrightarrow v_2 \longrightarrow v_3 \longrightarrow v_5 \rangle$ provided are not equal; according to the original definition, they do not satisfy the dominating relationship. Now, we relax this restriction. The number of services of $\langle s \longrightarrow v_1 \longrightarrow v_5 \rangle$ is 2, which is one less than $\langle s \longrightarrow v_2 \longrightarrow v_3 \longrightarrow v_5 \rangle$. If $\langle s \longrightarrow v_1 \longrightarrow v_5 \rangle$ reaches the same number of services of $\langle s \longrightarrow v_2 \longrightarrow v_3 \longrightarrow v_5 \rangle$, the number is 3, and it should add an edge. We assume the cost of the new added edge is ave_weight, that is the average edge weight of all the edges in the graph; as shown in the following, n is the number of edges and $weght_i$ is the i-th edge's weight:

$$ave_{weight} = \frac{\sum_{i=1}^{n} weight_i}{n}. \tag{12}$$

Now, after adding, we can find that route $\langle s \longrightarrow v_2 \longrightarrow v_3 \longrightarrow v_5 \rangle$ has smaller distance cost, so the $\langle s \longrightarrow v_2 \longrightarrow v_3 \longrightarrow v_5 \rangle$ is the dominating route, and $\langle s \longrightarrow v_1 \longrightarrow v_5 \rangle$ is dominated.

Definition 13 (optimize domination). Given a user's request $Q = \{q_1, \ldots, q_j\}$ and two partially explored candidate routes $R_1 = \langle s, v_1^1, \ldots, v_q^1 \rangle$ and $R_2 = \langle s, v_1^2, \ldots, v_q^2 \rangle$ $(1 \le q \le j)$, if $v_q^1 = v_q^2$ and the service number of R_1 is less than or equal to R_2, and $\text{cost}(R_1) + l * \text{ave\_weight} \le \text{cost}(R_2)$ holds, R_1 dominates R_2, denoted as $R_1 \prec_{QN} R_2$, where l is the difference value of the number of services of two routes.

For example, if R_2 have two services and R_1 have four services, in order to achieve the same number of services of R_1, we will add two average edge weights to R_2 as the route's estimated cost. After that, we decide the dominating relationship.

Lemma 14. *Given a query $(s, Q = \langle q_1, \ldots, q_j \rangle, k)$ and two partially explored routes R_1 and R_2, if $R_1 \prec_{QN} R_2$, then $cost(R_1^*) \le cost(R_2^*)$, where R_1^* and R_2^* are the optimal feasible routes that are extended from R_1 and R_2, respectively.*

Proof. Suppose $R_1 = \langle s, v_1^1, \ldots, v_q^1 \rangle$, $R_2 = \langle s, v_1^2, \ldots, v_q^2 \rangle$, $R_1^* = \langle s, v_1^1, \ldots, v_q^1, v_{q+1}, \ldots, v_j \rangle$, and $R_2^* = \langle s, v_1^2, \ldots, v_q^2, v_{q+1}, \ldots, v_i \rangle$. $R_1' = \langle v_q^1, v_{q+1}, v_{q+2} \ldots, v_j \rangle$ and $R_2' = \langle v_q^2, v_{q+2}', \ldots, v_i \rangle$ are the optimal route from R_1 and R_2, respectively, where v_{q+2} and v_{q+2}' have the same service and $v_q^1 = v_q^2$. According to Algorithm 2, $\cos t(v_q^2, v_{q+2}') \le \cos t(v_q^1, v_{q+1}, v_{q+2})$ and $\cos t(v_q^1, v_{q+1}, v_{q+2}) \le \cos t(v_q^2, v_{q+2}') + ave\_weight$, so $\cos t(R_1') \le \cos t(R_2') + l * ave\_weight$. Because of $\cos t(R_1^*) = \cos t(R_1) + \cos t(R_1')$ and $\cos t(R_2^*) = \cos t(R_2) + \cos t(R_2')$, we can know that $\cos t(R_1^*) \le \cos t(R_2^*)$. □

4.2.2. Priority Query. We introduce the priority of query in this section. By analyzing the travel routes of most users, we find that some services inherently have a higher priority than others. Next, we keep these services and their priorities in the dictionary D. For example, a large amount of data shows that users first go to the bank to withdraw money and then spend money, so the priority of withdrawing money is higher than that of consumption. If the services do not have the priority relationship, they are considered the same. By classifying the priority, the services with higher priority are queried firstly,

```
Input: Request: Q = ⟨r_1, r_2, ..., r_q⟩; N = |Q|: the number of services;
Priority dictionary: D(keyword, priority); initialize S=Q;
Output: priority set
1   n=1, HQ ⟵ Ø, QH ⟵ Ø, t=0, set ⟵ Ø, pre_set ⟵ Ø;
2   if |S|==N then
3      for r_n in Q do
4         if (Q- r_n)∩r_n.D ≠ Ø then
5            HQ= (Q- r_n)∩r_n.D
6            t=max(HQ.priority)
7            if t < r_n.priority then
8               r_n.priority=1
9            else
10              r_n.priority=0
11        else
12           r_n.priority=1
13           QH.add(r_n,r_n.priority)
14     for r_n in QH do
15        if r_n.priority==1 then
16           set.add(r_n)
17     pre_set⟵set;
18     S ⟵ Ø;
19  else
20     r'=pre_set-S;
21     HQ=(Q- r')∩r'.D;
22     if HQ ≠ Ø then
23        t=max(HQ.priority)
24        for r in HQ do
25           if r.priority==t then
26              S.add(r)
27     pre_set⟵S;
28     set⟵pre_set;
29  return set;
```

Algorithm 3: Priority(Q,D,S).

and the services with the same priority are queried randomly. We call this kind of query partial ordered query based on priority (POQP). In order to facilitate the classification of user request priority, we first plan the priority of services with obvious priority relationship in offline work.

At each iteration, the algorithm classifies the priority of user's requests and queries the high-priority requests firstly. Algorithm 3 assigns priority for query keywords and stores the result in a specific set named *pre_set* (lines 1). Initially, |S| is equal to N and we check every keyword in the Q. Some keywords have a prior relationship. If a keyword's priority level is the highest in these keywords, we define that the keyword's priority level is 1; otherwise, the priority is 0 (lines 2-10). For other keywords, they have no prior relationship, so they are independent. These keywords have no effect on other keywords, so we also define that their priority level is 1 (line 12). After that, we add the keyword to the result set and assign the result set to *pre_set* (lines 14-17). After the initial step, the algorithm finds the service with lower priority through the last finished service and stores the service into the result set (lines 19-28). Finally, the algorithm outputs the result set (line 29). Algorithm 3 is used in Algorithm 1. When the algorithm queries the nearest neighbor, Algorithm 3 can reduce the number of the candidate POIs.

By classifying the queries according to their priority, we can avoid the possibility that the route formed is inconsistent with the actual situation, and at the same time we can reduce the number of candidate POIs.

4.3. Route Replacement. We proposed a routing optimization mechanism, namely, route replacement, to check whether the routing cost can be shorter.

After obtaining the final route $R = \langle s, v_2, v_3, v_5, v_9 \rangle$ produced by Algorithm 1, we propose a postprocessing mechanism named route replacement to refine it. As shown in Figure 2, the route from v_5 to v_9 must go through v_7, and we can find that the services provided by v_2 are the same as those provided by v_7, and we can know that $dist(s, v_3)$ is smaller than the sum of $dist(s, v_2)$ and $dist(v_2, v_3)$. So, we can use v_7 to replace v_2. Finally, the shortest route is $R' = \langle s, v_3, v_5, v_7, v_9 \rangle$. Obviously, the routing cost of R' is smaller than that of R. When we refine the route, we should store these points which satisfies the following requirements: (1) those that have not been searched, (2) those that must be passed by the refined route, and (3) those that provide the same services as the existing points in the refined route. In the refining process, we justify whether to use these points to replace the existing

```
Input: route: R=⟨s, v1, v2, ..., v|R|⟩; query: Q
Output: route: R'
1 v0 ⟵ s, V ⟵ ∅
2 for vi in R and i < |R| do
3     V.add(between(vi,vi+1));
4 for v in V do
5   if QRP(v)∩Q ≠ Ø then
6     for vi in R do
7       if QRP(v)∩Q=QRP(vi)∩Q and cost(vi−1, vi+1)< cost(vi−1, vi)+cost(vi, vi+1) then
8         R'=R.del(vi);
9         R'=R'.add(v);
10      else continue;
11    else continue;
12 return R';
```

Algorithm 4: R2(Q,R).

point, which can reduce the route cost. We should choose the replacing point with more services that satisfies the user's requests, so it can not only reduce the cost but also make the route more concise by reducing the point in the route at the same time. We can understand the process of route replacement in detail through Algorithm 4.

As shown in Algorithm 4, the pseudocode has described the process of route replacement. Firstly, we define the notion that V is for storing POIs (line 1). These POIs are going to be used to replace the POI in the route. For each POI v_i in the route R, we look for the POIs between v_i and v_{i+1}. Then, we add these POIs to the V (lines 2-3). Next, we check each POI in the V and seek out the POI v that is able to meet the user's query (line 4-5). We look for a POI v_i in the route. If v_i and v are able to provide the same services, we compare the cost between $cost(v_{i-1}, v_{i+1})$ and $cost(v_{i-1}, v_i) + cost(v_i, v_{i+1})$. If $\text{cost}(v_{i-1}, v_{i+1})$ is less than $\text{cost}(v_{i-1}, v_i) + \text{cost}(v_i, v_{i+1})$, v_i is able to be replaced by v (lines 6-10).

5. Experimental Evaluation

5.1. Experimental Setup

5.1.1. Datasets. We use two real-world datasets from Zeng [26]. *Singapore* represents Foursquare check-in data collected in Singapore, and *Austin* represents Gowalla check-in data collected in Austin. *Singapore* has 189,306 check-in points, 5,412 locations, and 2,321 users. *Austin* has 201,525 check-ins, 6,176 locations, and 4,630 users. The same as suggested [26, 27], we built an edge between two locations if they were visited on the same date by the same user. The locations not connected by edges were ignored. We filled in the edge costs $t_{i,j}$ by querying the traveling time in minute using Google Maps API under driving mode. The statistic information of the dataset is shown in Table 6.

Both datasets were used in [26], which also studied a route planning problem. The datasets are not small considering the scenario for a daily trip in a city where the user has a limited cost budget. Even with 150 POIs to choose from, the number of possible routes consisting of 5 POIs can reach 70 billion. Compared to our work, Jeffrey [28] evaluated its itinerary recommendation methods using theme park data, where each park contains only 20 to 30 attractions.

Table 6: Dataset statistics.

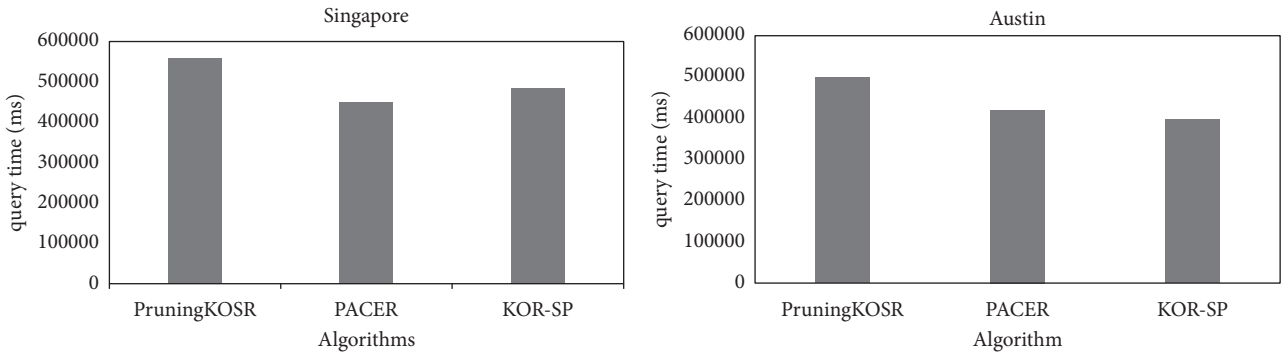
	#POI	#Edges
Singapore	1,625	24,969
Austin	2,609	34,340

5.1.2. Algorithms. We compared the following algorithms. *PACER* [8] models the personalized diversity requirement by retrieving POIs indexes related to feature space and route space, as well as various strategies of pruning search space with user preferences and constraints, and the optimal solution of the top-k path search problem is given. *PruningKOSR* [21] uses dominance relationship to filter temporarily unnecessary routes. *KOR-SP* is our proposed optimal algorithm.

5.1.3. Queries. For each KOR-SP query (s, k), we randomly select a source and an integer k, and then we issue query on all the graphs. In each experiment, 50 random query instances were constructed and the average query time was reported. If the query cannot be stopped within 4200 seconds or fails due to a memory overflow exception, we represent its corresponding query time as INF.

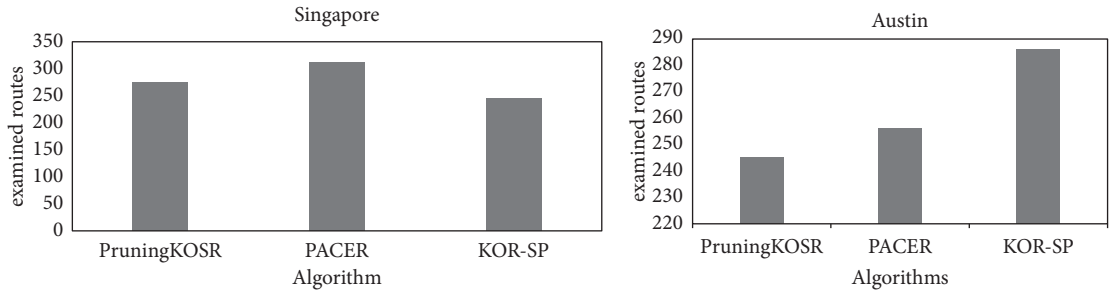
5.1.4. Evaluation Criteria. We evaluate the performance of different methods in four different aspects: the query runtime, the number of examined routes (witnesses), the number of (next) nearest neighbor (shortened as kNN) queries executed by calling Algorithm FindNN, and the cost of the routes.

5.2. Experimental Results. We first evaluate the efficiency of different algorithms answering KOR query in the default



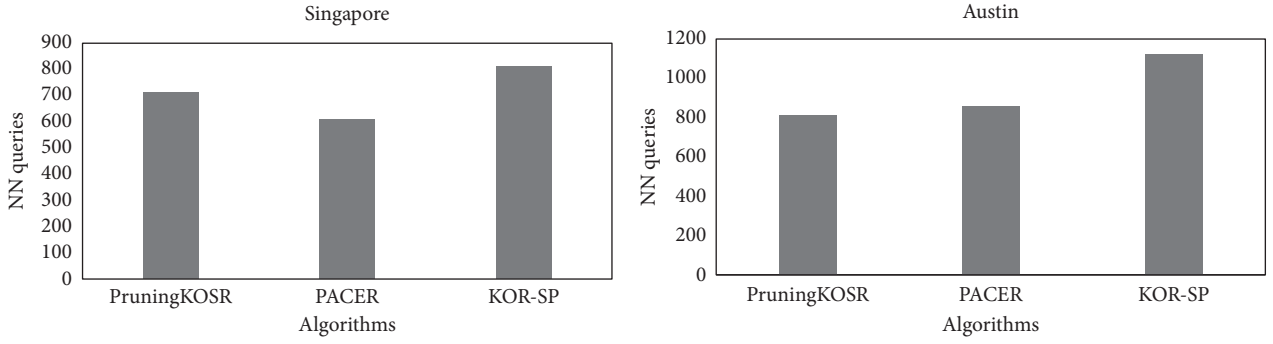
(a) methods on Singapore (b) methods on Austin

Figure 3: Run-time.



(a) methods on Singapore (b) methods on Austin

Figure 4: Examined routes.

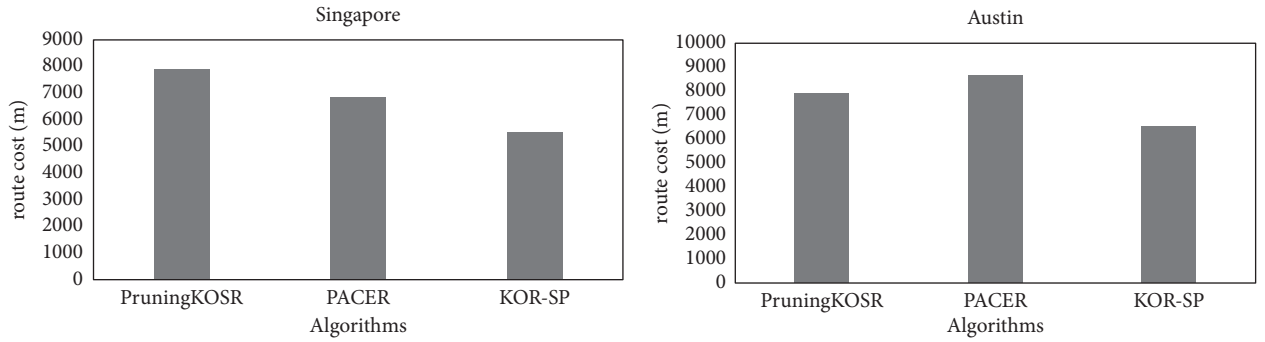


(a) methods on Singapore (b) methods on Austin

Figure 5: NN queries.

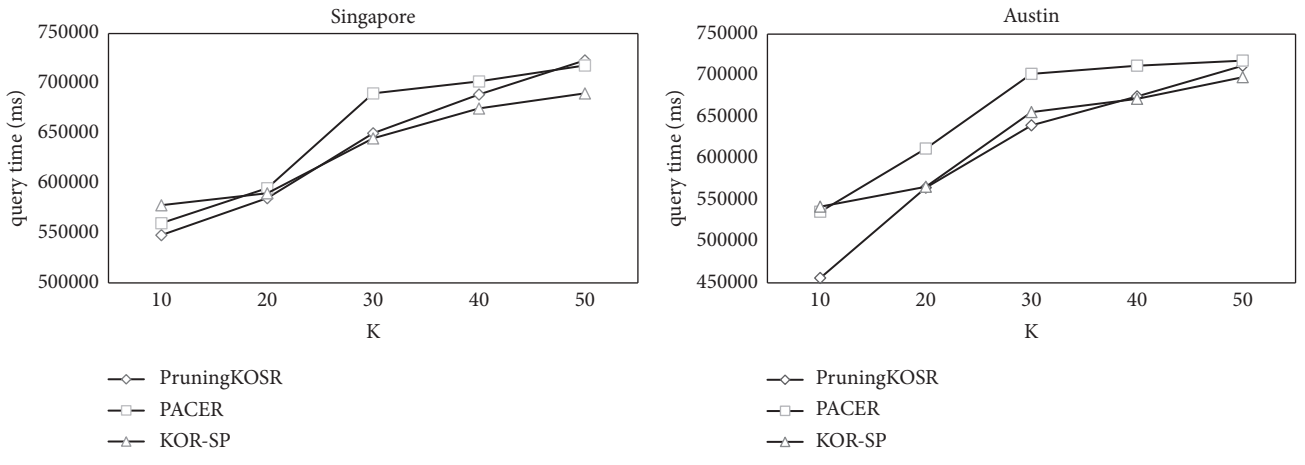
parameter setting on two real graphs and then evaluate the impact of parameters on the results.

5.2.1. Overall Performance under Default Parameter Settings. Figures 3–6 show the performance of three different algorithms on two graphs. The runtime of the algorithms on different graphs is displayed in Figure 3. Since all the algorithms have reduced the searching space, these can return the results on all graphs. At the same time, all the algorithms express efficient queries by using 2-hop label index. Figures 4 and 5 show the number of examined routes and NN queries, respectively. We can find that the number of examined routes in KOR-SP is much fewer than PruningKOSR on all graphs and the number of NN queries is larger than other algorithms. From this phenomenon, we can know the importance of a rich candidate. Because of semantic matching, KOR-SP has more candidate POIs and it can complete the route query with examining fewer routes. The consequence means KOR-SP is better than PruningKOSR. Figure 6 shows the cost of routes. As it is shown, the cost of routes of KOR-SP is much smaller than other algorithms, because this method has more



(a) methods on Singapore (b) methods on Austin

Figure 6: Route cost.



(a) Effect of k on Singapore (b) Effect of k on Austin

Figure 7: Running time vs.k.

candidate POIs that contains some POIs of which distance is shorter by semantic matching.

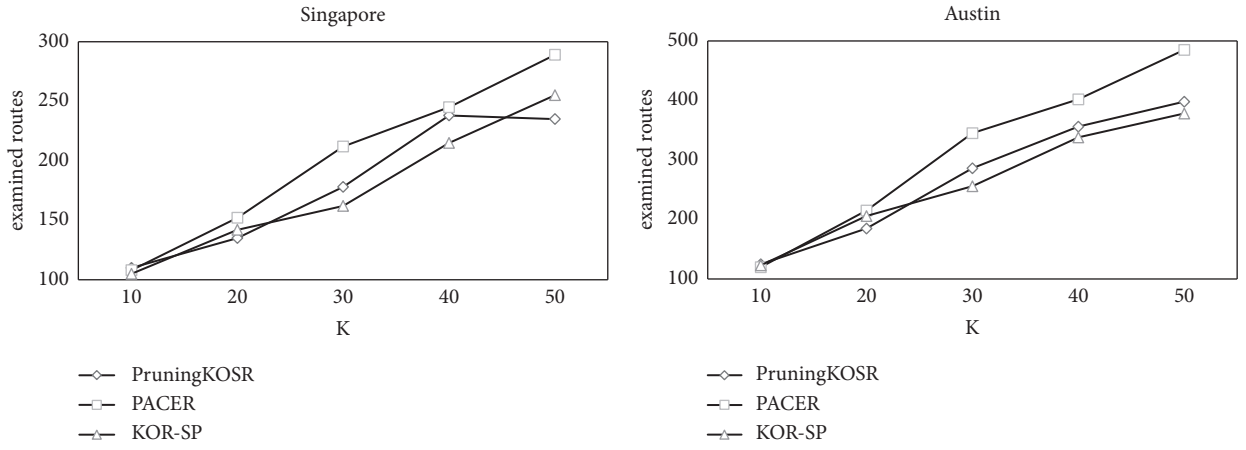
5.2.2. Effect k. Figures 7–10 show the influence of parameter k on the runtime of the three different methods on the two graphs. As shown in Figure 7, we know that all three methods can complete the function of query within the specified time, and with the increase of k, the advantages of KOR-SP are more and more obvious. Figures 8 and 9 show the impact of the number of routes and NN queries checked in different methods on different graphs. We can find that there are far fewer routes and NN queries checked in KOR-SP. Compared with other algorithms, this algorithm has significant advantages under different k conditions. Figure 10 shows the influence of k on routing cost. It can be seen from the figure that, due to semantic matching, routing cost of KOR-SP at different k is the lowest.

5.2.3. Effect β. Figure 11 shows the effect of parameter β. We can find that different β in KOR-SP algorithm has a great influence. The smaller the value of β is, the less the route cost is, because there are more candidate POIs with decreasing the β value, and then we can find more and more nearer neighbors to extend the route.

Figure 12 shows the difference among the four different KOR-SP algorithms. In the figure, the KOR-SP is the basic algorithm, the PKOR-SP is the KOR-SP combining with the priority relationship, the OKOR-SP is the KOR-SP combining with optimize domination, and RKOR-SP is KOR-SP combining with the route replacement. Form Figure 12(a), we can find that the PKOR-SP has the best performance. We know that the priority relationship helps us reducing the size of query. And the route replacement only refines the result of KOR-SP, so RKOR-SP's running time is longer than KOR-SP. From Figure 12(b), we can find that the OKOR-SP has the best performance. At the same time, other algorithms do not make much difference. By comparing other algorithms, we think that the optimized domination has a good effect.

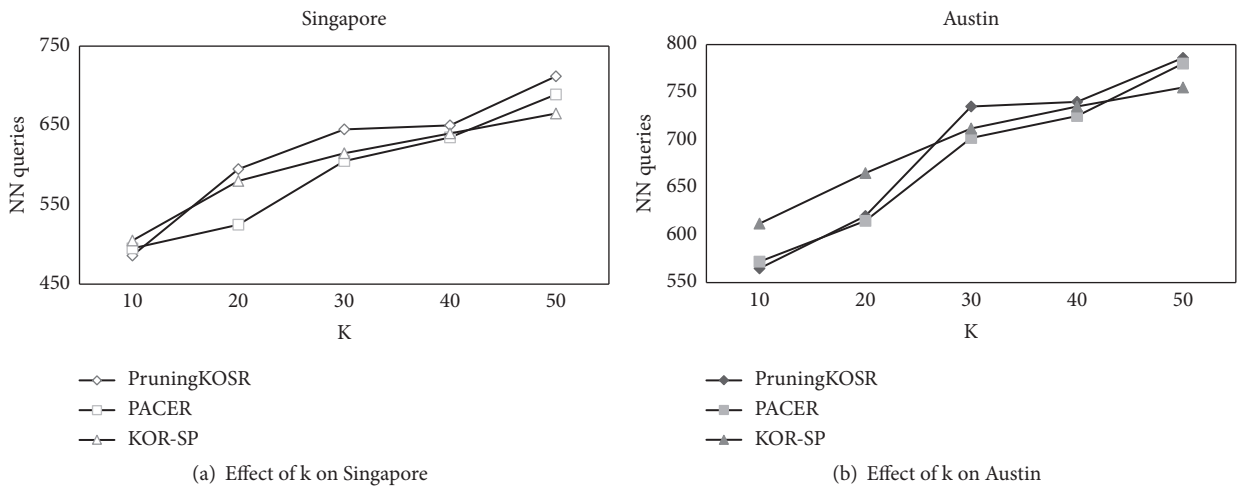
6. Conclusion

In this paper, we study the top-k optimal sequenced routes problem. We propose an efficient algorithm called KOR-SP, based on a novel route dominate relationship and a semantic



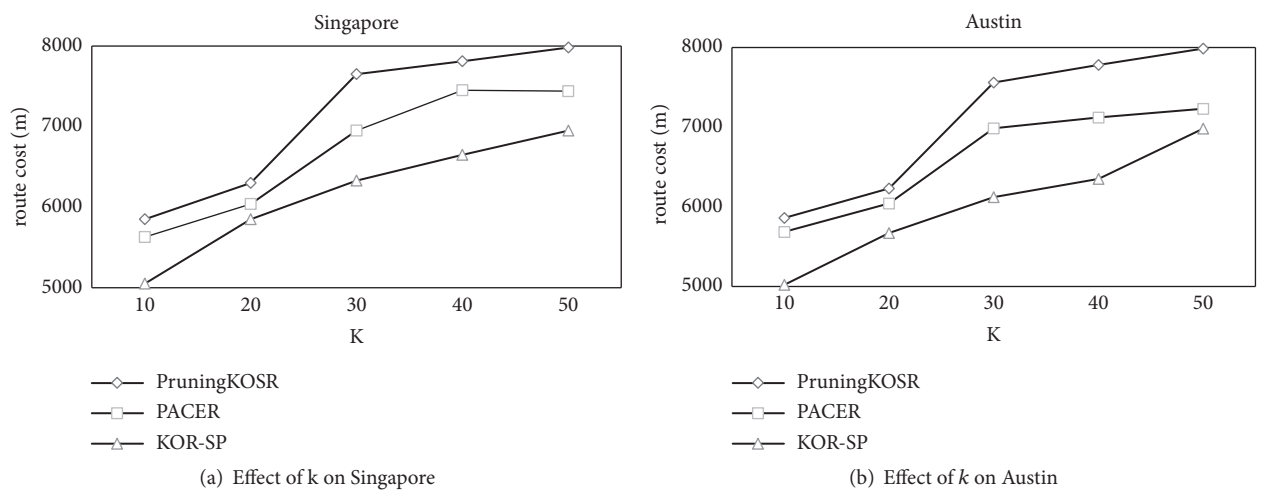
(a) Effect of k on Singapore (b) Effect of k on Austin

Figure 8: Examined routes vs.k.



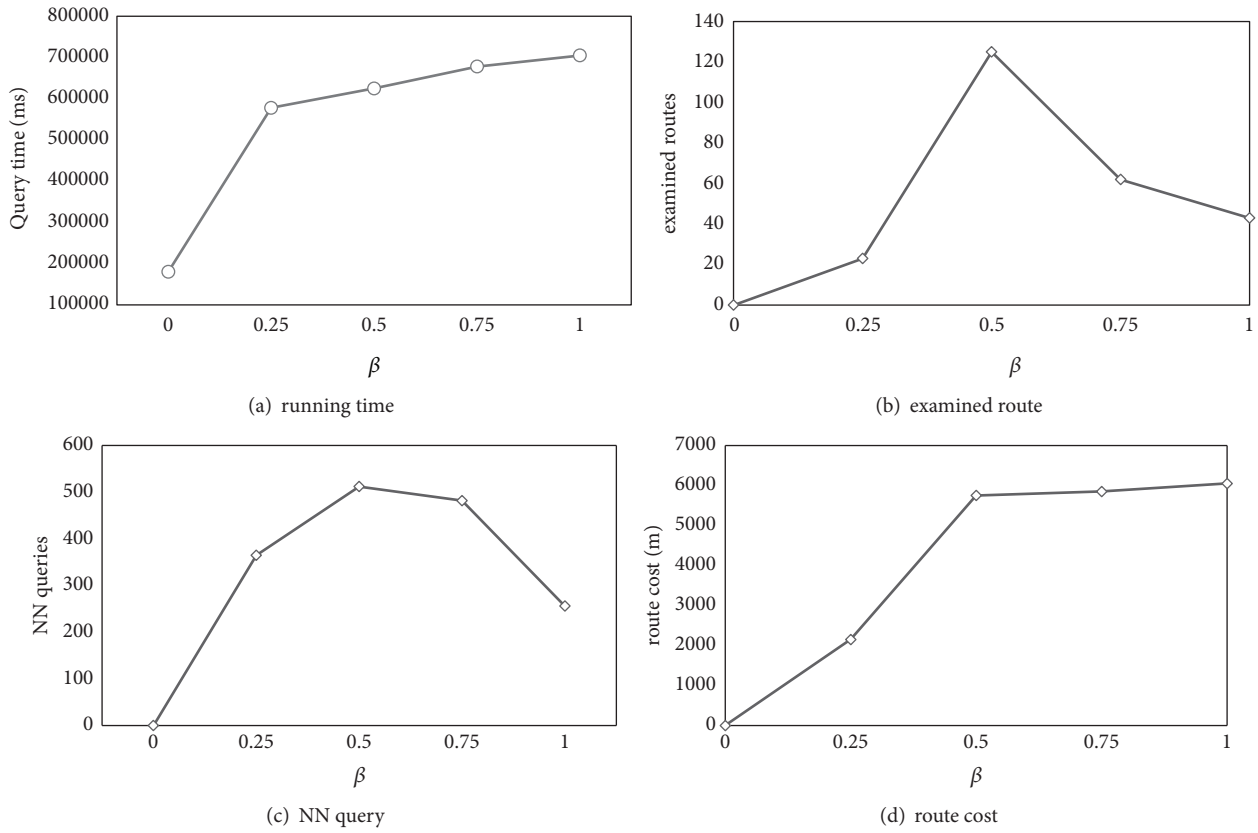
(a) Effect of k on Singapore (b) Effect of k on Austin

Figure 9: NN queries vs.k.



(a) Effect of k on Singapore (b) Effect of k on Austin

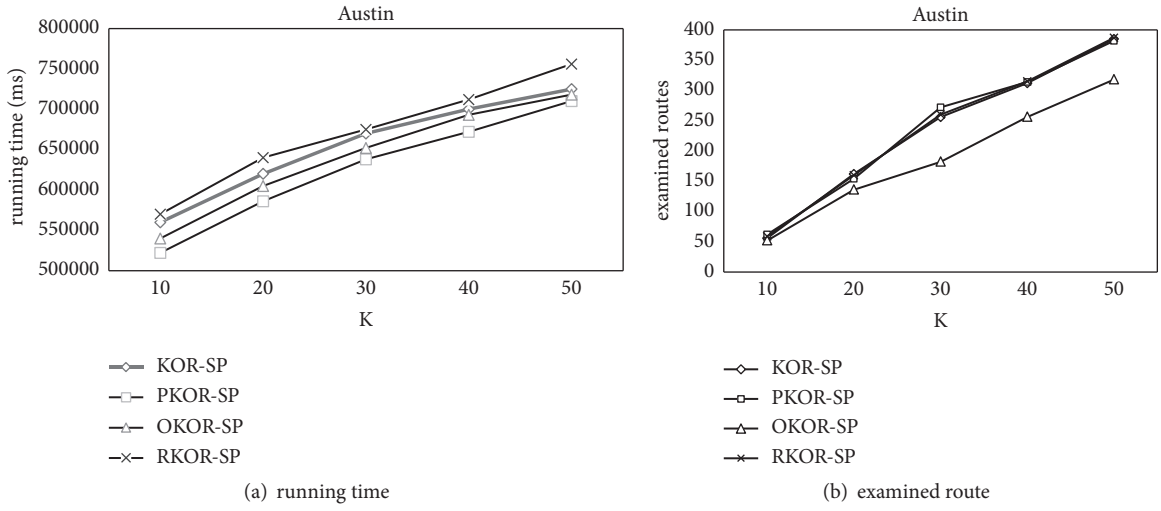
Figure 10: Route cost vs.k.



(a) running time (b) examined route

(c) NN query (d) route cost

Figure 11: Effect of β.



(a) running time (b) examined route

Figure 12: Performance on different KOR-SP algorithms in Austin.

matching by using the LDA model. Extensive experiments on real-world graphs demonstrate that the proposed algorithms are efficient. KOR-SP algorithm improves the flexibility of POI query and provides rich candidate sets for POI query by keyword semantic matching. And KOR-SP algorithm can quickly find the x-th nearest neighbor of the current POI by using FindNN algorithm and reduce the route search space by dominating relation. In addition, the algorithm uses route refinement mechanisms to improve route quality.

As a future work, we plan to study the keyword unordered query which is disordered for the whole, but it is order for part of keywords that have causality.

Data Availability

The graph data used to support the findings of this study are from [8, 26], and the datasets are available at https://github.com/LazyAir/SIGIR18.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research is partially funded by National Natural Science Foundation of China, under Grant no. 61602102 and no. 61872069, and the Fundamental Research Funds for the Central Universities, under Grant no. N161704004.

References

[1] S. H. Fang, E. H. Lu, and V. S. Tseng, "Trip recommendation with multiple user constraints by integrating point-of-interests and travel packages," in *Proceedings of the 2014 15th IEEE International Conference on Mobile Data Management (MDM)*, pp. 33–42, 2014.

[2] E. H. Lu, C. Lin, and V. S. Tseng, "Trip-Mine: an efficient trip planning approach with travel time constraints," in *Proceedings of the 2011 12th IEEE International Conference on Mobile Data Management (MDM)*, pp. 152–161, Lulea, Sweden, June 2011.

[3] J. Dai, C. Liu, J. Xu, and Z. Ding, "On personalized and sequenced route planning," *World Wide Web*, vol. 19, no. 4, pp. 679–705, 2016.

[4] F. Li, D. Cheng, M. Hadjieleftheriou, G. Kollios, and S.-H. Teng, "On trip planning queries in spatial databases," in *Proceedings of the 9th International Symposium on Spatial and Temporal Databases, SSTD 2005*, pp. 273–290, Brazil, August 2005.

[5] J. Eisner and S. Funke, "Sequenced route queries: getting things done on the way back home," in *Proceedings of the 20th ACM Sigspatial International Conference on Advances in Geographic Information Systems*, pp. 502–505, USA, 2012.

[6] Y. Ohsawa, H. Htoo, N. Sonehara, and M. Sakauchi, "Sequenced route query in road network distance based on incremental Euclidean restriction," *Database and Expert Systems Applications*, vol. 7446, no. 1, pp. 484–491, 2012.

[7] M. Sharifzadeh, M. Kolahdouzan, and C. Shahabi, "The optimal sequenced route query," *The VLDB Journal*, vol. 17, no. 4, pp. 765–787, 2008.

[8] H. Liang and K. Wang, "Top-k route search through submodularity modeling of recurrent POI features," in *Proceedings of the 41st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2018*, pp. 545–554, USA, July 2018.

[9] I. Abraham, D. Delling, A. V. Goldberg, and R. F. Werneck, "Hierarchical hub labelings for shortest paths," *Algorithms–ESA*, vol. 7501, pp. 24–35, 2012.

[10] T. Akiba, Y. Iwata, and Y. Yoshida, "Fast exact shortest-path distance queries on large networks by pruned landmark labeling," in *Proceedings of the 2013 ACM SIGMOD Conference on Management of Data*, pp. 349–360, USA, 2013.

[11] E. Cohen, E. Halperin, H. Kaplan, and U. Zwick, "Reachability and distance queries via 2-hop labels," *Siam Journal on Computing*, vol. 32, no. 5, pp. 937–946, 20032.

[12] R. Bramandia, B. Choi, and W. K. Ng, "On incremental maintenance of 2-hop labeling of large graphs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 5, pp. 682–698, 2010.

[13] J. J. Ying, W. Kuo, V. S. Tseng, and E. H. Lu, "Mining user check-in behavior with a random walk for urban point-of-interest recommendations," *ACM Transactions on Intelligent Systems and Technology*, vol. 5, no. 3, pp. 1–27, 2014.

[14] E. H.-C. Lu, W.-C. Lee, and V. S.-M. Tseng, "A framework for personal mobile commerce pattern mining and prediction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 5, pp. 769–782, 2012.

[15] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[16] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[17] E. M. Arkin and R. Hassin, "Approximation algorithms for the geometric covering salesman problem," *Discrete Applied Mathematics: The Journal of Combinatorial Algorithms, Informatics and Computational Sciences*, vol. 55, no. 3, pp. 197–218, 1994.

[18] F. Li, D. Cheng, M. Hadjieleftheriou, G. Kollios, and S. Teng, "On trip planning queries in spatial databases," in *Proceedings of the International Symposium on Spatial and Temporal Databases*, vol. 31 of *Lecture Notes in Computer Science*, no.1, pp. 273–290, Springer, Boston, MA, USA, 2005.

[19] K. Menger, "Ergebnisse eines mathematischen Kolloquiums," *Monatshefte Fur Mathematik - Monatsh Math*, vol. 39, no. 1, 1932.

[20] E. Ahmadi and M. A. Nascimento, "A mixed breadth-depth first search strategy for sequenced group trip planning queries," in *Proceedings of the 16th IEEE International Conference on Mobile Data Management*, pp. 24–33, USA, 2015.

[21] H. Liu, C. Jin, B. Yang, and A. Zhou, "Finding top-k optimal sequenced routes," *International Council for Open and Distance Education*, pp. 569–580, 2018.

[22] L. Tang, D. Cai, Z. Duan, J. Ma, M. Han, and H. Wang, "Discovering travel community for POI recommendation on location-based social networks," *Complexity*, vol. 2019, Article ID 8503962, 8 pages, 2019.

[23] J. Li, C. Liu, J. X. Yu, Y. Chen, T. Sellis, and J. S. Culpepper, "Personalized influential topic search via social network summarization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 7, pp. 1820–1834, 2016.

[24] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, no. 4-5, pp. 993–1022, 2003.

[25] M. Babenko, A. V. Goldberg, A. Gupta, and V. Nagarajan, "Algorithms for hub label optimization," *ACM Transactions on Algorithms (TALG)*, vol. 13, no. 1, pp. 1–17, 2016.

[26] Y. Zeng, X. Chen, X. Cao, S. Qin, M. Cavazza, and Y. Xiang, "Optimal route search with the coverage of users' preferences," in *Proceedings of the 24th International Joint Conference on Artificial Intelligence, IJCAI 2015*, pp. 2118–2124, Argentina, July 2015.

[27] X. Cao, L. Chen, G. Cong, and X. Xiao, "Keyword-aware optimal route search," *VLDB Endowment*, vol. 5, no. 11, pp. 1136–1147, 2012.

[28] K. H. Lim, J. Chan, S. Karunasekera, and C. Leckie, "Personalized itinerary recommendation with queuing time awareness," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2017*, pp. 325–334, Japan, August 2017.

Hindawi
Complexity
Volume 2019, Article ID 1532013, 12 pages
https://doi.org/10.1155/2019/1532013

Research Article

Incremental Bilateral Preference Stable Planning over Event Based Social Networks

Boyang Li,[1] **Yurong Cheng**,[2] **Guoren Wang**,[2] **and Yongjiao Sun**[1]

[1]*School of Computer Science and Engineering, Northeastern University, Shenyang 110819, China*
[2]*School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China*

Correspondence should be addressed to Yurong Cheng; yrcheng@bit.edu.cn

Received 24 January 2019; Accepted 2 April 2019; Published 16 April 2019

Guest Editor: Jiajie Xu

In recent years, Event Based Social Networks (EBSNs) appear in people's daily life and are becoming increasing popular. In EBSNs, one typical task is to make personalized plans for users. Existing studies only consider the preference of users. They make plans by selecting interesting events for users. However, for organizers of events, they also would like more high-quality users to participate in their events, which may make the events more exciting. Existing studies are user-centered and ignore the requirement of organizers. What is more, organizers are allowed to modify their events dynamically before they are held. The platforms should be able to dynamically adjust the schedules of users. Therefore, we identify a new Incremental Bilateral Preference Stable Planning (IBPSP) problem over EBSNs and propose several solutions to deal with different situations. We conduct extensive experiments to verify the efficiency and effectiveness of the proposed algorithms.

1. Introduction

In recent years, Event Based Social Networks (EBSNs) [1] have experienced rapid development and attracted much attention from both industry and academia fields. EBSNs, such as Meetup (https://www.meetup.com/) and Plancast (http://plancast.com/), link the online social groups and the offline events. Taken Meetup(https://www.meetup.com/) as an example, it has attracted more than 16 million users with more than 300 thousand events held each month.

In EBSNs, one typical task is to select suitable events and make personalized plans for users to participate in according to the labels that users select as their interested points. Therefore, we can evaluate the interest of users to events based on the similarity of labels of users and events, called utility score [2–6]. The higher the utility score is, the more interested the user is to the event. Besides, the spatial distance is another import factor in personalized planning; the total travel cost of a plan should not be more than the travel budget of the user. Moreover, a user may participate in more than one events; the platform should guarantee that the time periods of events that the user participate in are not overlapped. Therefore, the goal is to make plans for all the users to maximize the utility score where the travel cost of each user cannot be more than his budget and events in the same plan do not conflict with each other.

Example 1. Suppose we have 5 users and 3 events, and details are shown in Table 1. The first row describes users and their travel budget, and the first column gives the events and the upper user bound. The last column is the time period when each event is held, and the time period of e_2 and e_3 are overlapped. It means that users cannot participate in both e_1 and e_3. The other columns are the utility scores of each user. The locations of users and events are shown in Figure 1. The locations are described in a 2D space and the distance between any two locations is Euclidean distance. For a user, he starts from his current location, participates in each event in his plan one by one, and goes back to his start location. For example, the purple lines are the plan of u_2. He starts from $(8, 2)$, participates in e_3 and e_1, and go back to $(8, 2)$ at last. The travel cost is 16.9, which is no more than the travel budget. The total utility of all users whose plans are shown in Figure 1 is 3.7.

Table 1: Users' utility to each event.

	$u_1(40)$	$u_2(40)$	$u_3(10)$	$u_4(52)$	$u_5(18)$	Time
$e_1(3)$	0.6	0.3	0.4	0.5	0.2	12:30-14:00
$e_2(1)$	0.1	0.4	0.8	0.6	0.7	10:00-12:00
$e_3(2)$	0.1	0.7	0.9	0.2	0.9	09:00-12:00

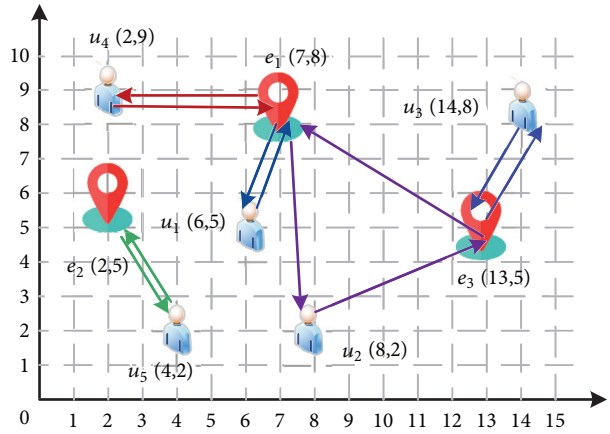


Figure 1: The location of users and events.

Table 2: Events' utility to each user.

	$e_1(3)$	$e_2(1)$	$e_3(2)$
$u_1(12)$	0.3	0.4	0.8
$u_2(40)$	0.1	0.8	0.7
$u_3(10)$	0.6	0.4	0.6
$u_4(52)$	0.2	0.7	0.5
$u_5(10)$	0.9	0.5	0.9

Existing studies [2–6] only focus on the utility of users. However, the event organizers also have preference towards users for holding the event successfully. For example, organizers prefer users who are more influential in the social network to participate in their events, which may improve the quality of the events. Therefore, the personalized planning problem over EBSNs should be a bilateral planning problem among users and events. The existing techniques cannot give plans that satisfy both users and event organizers. For Example 1, we give the utility scores of events to users in Table 2. u_1, u_2, and u_4 are the worst three users of e_1. e_2 prefers u_4 than u_5. u_2 and u_3 are not the best users of e_3. Through analysis, these methods cannot consider the preference of both users and events at the same time.

Based on the aforementioned example, it is more reasonable to consider bilateral preference of users and events than to only consider the preference of users in real applications. Moreover, the events may be modified for some reasons; the plans for users should be updated quickly once it happens. Therefore, we formally define a new personalized planning problem in EBSNs. Different from existing studies, this problem is a bilateral planning problem which considers the preference of both users and events instead of only considering the preference of user and the attributes of user and events change dynamically. In summary, to solve this problem, we make the following contributions:

(i) We identify a new Incremental Bilateral Preference Stable Planning (IBPSP) problem over EBSNs, which aims to dynamically make plans satisfying both users and event organizers.

(ii) We present solutions to give stable plans in different situations as the attributes of users and events change.

(iii) We verify the effectiveness and efficiency of the proposed methods with extensive experiments on real dataset.

The rest of this paper is organized as follows. Section 2 briefly introduces some basic concepts and formalizes the problem. In Section 3, we give the solutions for different conditions. Extensive experiments are conducted on a series of real-life datasets and the performance is evaluated in Section 4. Section 5 discusses the related works. Finally, we give our conclusion in Section 6.

2. Problem Statement

In this section, we introduce some basic concepts of EBSNs firstly. And then, we formally define the Incremental Bilateral Preference Stable Planning (IBPSP) problem. Symbols used in our paper are shown in Table 3.

Given an event based social network with n users and m events, we denote U as the user set and E as the event set. Each user $u_i \in U$ is described by a 2-tuple $\langle \mathbf{l}_{u_i}, B_i \rangle$, where $\mathbf{l}_{u_i}$ is the location of u_i in a 2D space, and B_i is the travel cost budget. Each event $e_j \in E$ is described by a 4-tuple $\langle \mathbf{l}_{e_j}, \eta_j, t_j^s, t_j^e \rangle$, where $\mathbf{l}_{e_j}$ is the location of e_j in a 2D space, η_j is the largest

Table 3: Summary of symbols used in our paper.

Symbol	Description
E	The event set
e	An event
U	The user set
u	A user
$\mathbf{l}_{u_i}$	Location of u_i
B_i	The travel cost budget of u_i
$\mathbf{l}_{e_j}$	Location of e_j
η_j	The largest number of participants of e_j
t_j^s	The start time of e_j
t_j^e	The end time of e_j
D_i	The travel cost of u_i
$p_u(u_i, e_j)$	The utility score of u_i to e_j
$p_e(e_j, u_i)$	The utility score of e_j to u_i
$PU(u_i)$	The preference rank of u_i to all the events
$PE(e_j)$	The preference rank of e_j to all the users
$\mathscr{P}$	The global plan
$\mathscr{P}(u_i)$	The set of events that u_i will participate in
$\mathscr{P}(e_j)$	The set of users who will participate in e_j

number of participants, and t_j^s and t_j^e are the start time and end time of e_j. For user u_i, the utility score to event e_j is denoted as $p_u(u_i, e_j)$, and the utility score of e_j to u_i is denoted as $p_e(e_j, u_i)$, where $p_u(u_i, e_j), p_e(e_j, u_i) \in [0, 1]$. We denote the preference rank of u_i is $PE(u_i)$, and the preference rank of e_j is $PE(e_j)$. The order of utility scores is strict; that is, no two utility scores are equal.

A global plan $\mathscr{P}$ is a set of plans for all the users in the platform, denoted as $\mathscr{P} = \{(u_i, e_j) \mid 1 \le i \le n,\ 1 \le j \le m\}$. Events in each user's plan cannot conflict with each other. In other words, for two events e_k and e_h in the same user's plan, if e_k starts before e_h, e_k should end earlier than e_h. A user may participate in several events according to his plan; the travel cost is the sum of the distance that he travels. The distance between users and events is Euclidean distance.

Definition 2 (blocking pair). For a user u_i and an event e_j, where $(u_i, e_j) \notin \mathscr{P}$ and the travel budget is enough to participate in e_j, if u_i prefers e_j to at least one event in his plan and e_j prefers u_i to at least one user that will participate in e_j, then (u_i, e_j) is a *blocking pair*.

Due to the existing of travel budget, the travel cost may be larger than the budget of u_i after the platform arrangements u_i participate in e_j. In this case, (u_i, e_j) is not a blocking pair.

Back to Example 1, according to the plans in Figure 1 and the utility scores in Tables 1 and 2, (u_4, e_2) is a blocking pair. The reason is that u_4 prefers e_2 to e_1, and e_2 prefers u_4 to u_5. (u_5, e_3) is not a blocking pair. The reason is that the travel budget of u_5 is not enough to travel to e_3, though they prefer each other. Figure 2 shows planning results without the blocking pairs.

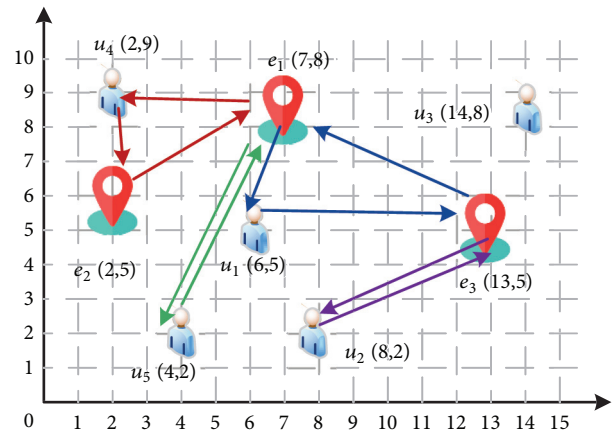


Figure 2: A planning without blocking pairs.

After defining the above concepts, we next introduce the IBPSP problem as follows.

Definition 3 (IBPSP problem). Given a set of users U, a set of events E, the preference rank PE, and PU over an EBSN platform, which allows the attributes of users and events change dynamically, the IBPSP problem is to find a global plan $\mathscr{P}$ for all the users such that the following constraints are satisfied:

(i) Events in the same user's plan do not conflict with each other, where $\forall_i\ \forall e_k \ne e_h \in \mathscr{P}(u_i),\ t_{e_k}^s < t_{e_h}^s \Longrightarrow t_{e_k}^e < t_{e_h}^e$.

(ii) The travel cost of users is not more than their travel budget, where $\forall_i\ D_i \le B_i$.

Input: PE, PU, E, U, $\mathscr{P}_{old}$, u_i
Output: $\mathscr{P}$
1 $\mathscr{P} \longleftarrow \mathscr{P}_{old}$
2 $E_{ac} \longleftarrow \emptyset$
3 **While**($D_i > B_i$)
4 $e_j \longleftarrow$ event with the worst utility score in $\mathscr{P}(u_i)$
5 Put e_j into E_{ac}, remove e_j, u_i from $\mathscr{P}(u_i)$, $\mathscr{P}(e_j)$
6 **EndWhile**
7 **Return** Update_event_plan(PE, PU, E, U, $\mathscr{P}_{old}$, E_{ac})

Algorithm 1: Budget decreasing algorithm.

(iii) The number of participants is not more than the upper bound of events, where $\forall_j \, |\{(u_i, e_j) \in \mathscr{P}, \ 1 \leq i \leq n\}| \leq \eta_j$.

(iv) The global plan does not contain any blocking pairs.

(v) The plan is updated when the attributes change.

In the IBPSP problem, both users and events do not prefer other events and users that are out of the plan, it will reach a stable state. From the perspective of economics, in the existence of competitive relations, the stable state is the most reasonable and the most consistent with the law of development of things [7]. In practice, information of events and profiles of users are subject to change. Thus, a reasonable event planning system should support incremental updates.

3. Solutions to IBPSP Problem

In this section, we first introduce which attributes of users and events may change. And then, we propose several solutions to deal with the changes and update the plans that are made based on the former attributes.

3.1. Changes Caused by Users and Events

Travel Budget of Users. Users can modify their travel budget according to their own intentions. For example, if a user has enough to enjoy the events, he may increase the travel budget to travel further or participate in more events. A user may also travel a shorter distance or even cancel the travel plan due to the bad weather or fall ill.

Participation Upper Bounds of Events. The maximum capacity of the events is limited by a number of factors. For example, if the event is more popular than expected, the organizers may choose a larger venue and increase the upper bound of participations. The opposite, however, is also possible. If a tourism interest group organizes a visit to a scenic spot, the organizer may reduce the number of people due to the restricted traffic conditions of the scenic spot.

Start Times and Times of Events. The organizer may modify the start or end time if the venue is occupied during the time period when the event will be held or some important guests cannot participate in the event on time.

Events Are Added or Cancelled. In EBSNs, new events will be added at any time. When a new event is added, the platform needs to arrange suitable users to participate in this event. The planned events can also be cancelled for all kinds of reasons. When it happens, the platform needs to arrange users to participate in other events that are suitable for them.

3.2. Solutions.

In this section, suppose the platform has made plans for users. And then, the changes which are introduced above come. We provide solutions on how to solve the changes when the changes happen.

3.2.1. B_i Is Decreased. When the travel budget of a user decreases, it may lead to that he cannot participate in some events that are planned by the platform for him. The platform needs to remove some events from his plan until the total travel cost is no more than the budget. What is more, the plans also need to be adjusted to make sure that there are no blocking pairs in the new plan.

The pseudocode is illustrated in Algorithm 1. When the travel budget of u_i changes, we first initialize a set E_{ac} as an empty set (Line 2). E_{ac} stores the events that are affected by the change. While the travel cost is larger than the travel budget, we find an event e_j in $\mathscr{P}(u_i)$ whose utility score is the lowest in $\mathrm{PU}(u_i)$, remove it from $\mathscr{P}(u_i)$, and put it into E_{ac} (Lines 3-6). After this step, we make sure that the travel cost is not greater than the new travel budget. Since u_i cannot participate in some events, there may be blocking pairs in the plans of these events. Finally, Algorithm 2 is called to update the plans of events in E_{ac}.

Algorithm 2 terminates until E_{ac} is empty and returns a new global plan. In each loop, we pop an event e_j from E_{ac} (Line 3) and start to enumerate each user u_i whose utility is not greater than the lowest utility in current plan $\mathscr{P}(e_j)$ until the number of participants equals η_j (Lines 4-5). If e_j conflicts with e_k that has a higher utility in (u_j), then we start again from the next user (Lines 7-9). Otherwise, we remove the events whose utility is less than e_j from $\mathscr{P}(u_i)$ and put them in a conflict set and they are also put into E_{ac} (Lines 10-12). Then we can add e_j to $\mathscr{P}(u_j)$ (Line 13). If the travel cost

Input: PE, PU, E, U, $\mathscr{P}_{old}$, E_{ac}
Output: $\mathscr{P}$
1 $\mathscr{P} \longleftarrow \mathscr{P}_{old}$
2 **While**($E_{ac} \neq \emptyset$)
3 Pop e_j from E_{ac}
4 $u_h \longleftarrow$ user with the worst utility in $\mathscr{P}(e_j)$
5 **Foreach** (u_i after u_h in $PE(e_j) \bigwedge$ the number of participants does not exceed η_j)
6 $conflict\_events = \phi$
7 **If**($\exists e_k \in \mathscr{P}(u_j)$ that conflicts with e_j)
8 **continue**
9 **EndIf**
10 **Foreach**($e_k \in \mathscr{P}(u_j)$)
11 Remove e_k from $\mathscr{P}(u_i)$, put e_k into $conflict\_events$, put e_k into E_{ac}
12 **EndFor**
13 Put e_j into $\mathscr{P}(u_j)$
14 **While**($D_i > B_i$)
15 Remove the worst event e_k from $\mathscr{P}(u_i)$
16 **If**(e_j is removed)
17 **break**
18 **EndIf**
19 Put e_k into $conflict\_events$, put e_k into E_{ac}
20 **EndWhile**
21 **If**(e_j has been removed from $\mathscr{P}(u_i)$)
22 **Foreach**($e_k \in conflict\_events$)
23 Put e_k back to $\mathscr{P}(u_i)$, remove e_k from E_{ac}
24 **EndFor**
25 **EndIf**
26 **EndFor**
27 **Endwhile**
28 **Return** $\mathscr{P}$

Algorithm 2: Update_event_plan.

is greater than the travel budget, we remove some events and put them into the conflict set and E_{ac} (Lines 14-20). If e_j is also removed in the former step, it means e_j is too fat for u_j to participate in, and $\mathscr{P}(u_i)$ is rehabilitated (Lines 21-25).

Example 4. Based on the planning of Figure 2, suppose that the travel budget of u_1 decreases to 5. Then, e_3 is removed from $\mathscr{P}(u_1)$ and put into E_{ac}. We pop e_3 from E_{ac}, and u_3 can participate in it; a new stable planning is constructed.

3.2.2. η_j Is Decreased. When η decreases, the number of participants may exceeds the new upper bound in current plans. The platform needs to remove some participants and adjust the plans of these participants.

The pseudocode is illustrated in Algorithm 3. When η_j changes, we first initialize a set U_{ac} as an empty set (Line 2). U_{ac} stores the users that are affected by the change. While the number of participants is larger than the new upper bound, we find a user u_i in $\mathscr{P}(e_j)$ whose utility score is the lowest in $PE(e_j)$, remove him from $\mathscr{P}(e_j)$, and put it into U_{ac} (Lines 3-6). After this step, we make sure that the new η is satisfied. Finally, Algorithm 4 is called to update the plans of users in U_{ac}.

Algorithm 4 terminates until U_{ac} is empty and return a new global plan. In each loop, we pop a user u_i from U_{ac} (Line 3) and start to enumerate each event e_j whose utility are not greater than the lowest utility in current plan $\mathscr{P}(u_i)$ (Lines 4-5). If e_j conflicts with e_k that has a higher utility in $\mathscr{P}(u_i)$, then we start again from the next event (Lines 6-8). Otherwise, add e_j, u_i into $\mathscr{P}(u_i)$, $\mathscr{P}(e_j)$ (Line 9). If the number of participants exceeds η_j, we remove the user with the lowest utility in $\mathscr{P}(e_j)$ and put the user into U_{ac} if he is not u_i (Lines 9-18).Finally the new global plan $\mathscr{P}$ is returned (Line 21).

Example 5. Based on the planning of Figure 2, suppose that the participant bound of e_3 decreases to 1. Then, e_3 is removed from the planning of u_2, and u_2 is put into U_{ac}. We pop u_2 from U_{ac}, then e_2 is removed from the planning of u_4 and u_4 is put into U_{ac}. u_4 cannot participate in other events; a new stable planning is constructed.

3.2.3. t_j^s or t_j^e Is Changed. If the start or end time changes, it may cause many conflicts in the original plan $\mathscr{P}$. There are four kinds of cases that the global plan needs to be updated: (1) the new event conflicts with events that with higher utility

Input: $PE, PU, E, U, \mathscr{P}_{old}, e_j$
Output: $\mathscr{P}$
1 $\mathscr{P} \longleftarrow \mathscr{P}_{old}$
2 $U_{ac} \longleftarrow \emptyset$
3 **While**($|\mathscr{P}(e_j)| > \eta_j$)
4 $u_i \longleftarrow$ user with the worst utility score in $\mathscr{P}(e_j)$
5 Put u_i into U_{ac}, remove e_j, u_i from $\mathscr{P}(u_i)$, $\mathscr{P}(e_j)$
6 **Endwhile**
7 **Return** Update_user_plan($PE, PU, E, U, \mathscr{P}_{old}, U_{ac}$)

Algorithm 3: η decreasing algorithm.

Input: $PE, PU, E, U, \mathscr{P}_{old}, U_{ac}$
Output: $\mathscr{P}$
1 $\mathscr{P} \longleftarrow \mathscr{P}_{old}$
2 **While**($U_{ac} \neq \emptyset$)
3 Pop u_i from U_{ac}
4 $e_k \longleftarrow$ event with the worst utility in $\mathscr{P}(u_i)$
5 **Foreach** (e_j after e_k in $PU(u_i)$)
6 **If**($\exists e_h \in \mathscr{P}(u_i)$ conflicts with e_j)
7 **break**
8 **EndIf**
9 Add e_j, u_i into $\mathscr{P}(u_i)$, $\mathscr{P}(e_j)$
10 **If**(the number of participants exceeds η_j)
11 **If**(u_i is the worst user in $\mathscr{P}(e_j)$)
12 Remove e_j, u_i from $\mathscr{P}(u_i)$, $\mathscr{P}(e_j)$
13 **break**;
14 **ElseIf**
15 Remove the worst user u_k from $\mathscr{P}(e_j)$
16 Put u_k into U_{ac}
17 **EndIf**
18 **EndIf**
19 **EndFor**
20 **Endwhile**
21 **Return** $\mathscr{P}$

Algorithm 4: Update_user_plan.

scores in users' plans, (2) the event does not conflict with events with higher utility scores in the same plan, however, some events with lower utility scores that conflict with it, (3) the original event conflicts with some events in the users' plan but the conflicts do not exist after the change, and (4) some events out of users' conflict with the original event but the conflicts do not exist after the change.

The pseudocode is illustrated in Algorithm 5. We initialize U_{ac} and E_{ac} at first (Line 1). And then, we find all the users whose plans are affected by the change, remove e_j from their plans, and put them into U_{ac} (Lines 3-5). The plan of e_j is cleared and needs to be replanned (Line 6). For users in U_{ac} and events in E_{ac}, we call Algorithms 2 and 3 to update their plans (Lines 7-12).

3.2.4. The Other Situations. We discuss how to solve the other situations, based on the above algorithms.

B_i Is Increased. When the travel budget of a user increases, the user can participate in more events. We can update his plan by putting him into U_{ac} and calling Algorithm 4. However, we need to make a small change to Algorithm 4. We cannot enumerate events starting from whose utility is not greater than the lowest utility in current plan; it will result in blocking pairs. The user may participate in events with higher utility than events in the current plan, because his travel budget may be large enough to travel to the locations of these events. Therefore, we enumerate events starting from whose utility is the largest and add them into the plan if possible.

Input: PE, PU, E, U, $\mathcal{P}_{old}$, e_j
Output: $\mathcal{P}$
1 $U_{ac} = \phi, E_{ac} = \emptyset$
2 $\mathcal{P} \longleftarrow \mathcal{P}_{old}$
3 **Foreach**($u_i \in \mathcal{P}(e_j)$)
4 Remove e_j from $\mathcal{P}(u_i)$, put u_i into U_{ac}
5 **EndFor**
6 Empty $\mathcal{P}(e_j)$, put e_j into E_{ac}
7 **If**($U_{ac} \neq \emptyset$)
8 $\mathcal{P} \longleftarrow$Update start users($PE$, PU, E, U, $\mathcal{P}$, U_{ac})
9 **EndIf**
10 **If**($E_{ac} \neq \emptyset$)
11 $\mathcal{P} \longleftarrow$Update start events($PE$, PU, E, U, $\mathcal{P}$, E_{ac})
12 **EndIf**
13 **Return** $\mathcal{P}$

Algorithm 5: t_j^s or t_j^e changing algorithm.

Input: PE, PU, E, U, $\mathcal{P}_{old}$, e_j, u_i
Output: $\mathcal{P}$
1 $U_{ac} = \phi, E_{ac} = \emptyset$
2 $\mathcal{P} \longleftarrow \mathcal{P}_{old}$
3 **While**($D_i > B_i$)
4 $e_j \longleftarrow$ event with the worst utility score in $\mathcal{P}(u_i)$
5 Put e_j into E_{ac}, remove e_j, u_i from $\mathcal{P}(u_i)$, $\mathcal{P}(e_j)$
6 **Endwhile**
7 **Foreach**($u_i \in \mathcal{P}(e_j)$)
8 Remove e_j from $\mathcal{P}(u_i)$, put u_i into U_{ac}
9 **EndFor**
10 Empty $\mathcal{P}(e_j)$, put e_j into E_{ac}
11 **If**($U_{ac} \neq \emptyset$)
12 $\mathcal{P} \longleftarrow$Update start users($PE$, PU, E, U, $\mathcal{P}$, U_{ac})
13 **EndIf**
14 **If**($E_{ac} \neq \emptyset$)
15 $\mathcal{P} \longleftarrow$Update start events($PE$, PU, E, U, $\mathcal{P}$, E_{ac})
16 **EndIf**
17 **Return** $\mathcal{P}$

Algorithm 6: Multiple changes in one run.

η_j *Is Increased*. When η increases, the event can accommodate more participants; we then put it into E_{ac} and call Algorithm 2 to update the plans. Note that if the number of participants is less than the original upper bound, the plans need not to be update.

3.2.5. Multiple Changes in One Run. We consider a more complex case in which multiple changes come at the same time. As introduced above, all the changes can be solved by two basic algorithms. The main idea is that we find the events and users whose plans need to be updated and conduct Algorithms 2 and 4 to update their plans.

The pseudocode is illustrated in Algorithm 6. We initialize U_{ac} and E_{ac} at first (Line 1). While the travel cost is larger than the travel budget, we find an event e_j in $\mathcal{P}(u_i)$ whose utility score is the lowest in $\mathrm{PU}(u_i)$, remove it from $\mathcal{P}(u_i)$, and put it into E_{ac} (Lines 3-6). And then, we find all the users whose plans are affected by the change, remove e_j from their plans, and put them into U_{ac} (Lines 7-9). The plan of e_j is cleared and needs to be replanned (Line 10). For users in U_{ac} and events in E_{ac}, we call Algorithms 2 and 3 to update their plans (Lines 11-17).

4. Performance Evaluation

In this section, we provide an empirical evaluation of our proposed algorithms, including the experimental environment, algorithm running time, memory cost, and the total sum of utility scores of users.

4.1. Experiment Environment and Dataset. We conduct our algorithms over a real-life dataset, the Meetup dataset [1],

Table 4: Real datasets.

City	$\|U\|$	$\|E\|$	Mean of η	Conflict ratio
Beijing	113	16	50	0.25
Auckland	569	37	50	0.25
Hawaii	2967	817	50	0.25
Hong Kong	3528	1324	50	0.25
Singapore	9893	4257	50	0.25
Vancouver	16095	11536	50	0.25

Table 5: Synthetic datasets.

	Value
$\|U\|$	200, 500, 1000, **5000**
$\|E\|$	200, 500, **1000**, 5000

which is a popular event based social network. The dataset records the locations of users and the tags that indicate the interesting points of the users. It also records the locations of events where they are held. The events are held by interesting groups; therefore, the tags of events are the tags of whom hold them. The utility scores are calculated as the method in [1]. We extract six datasets of different cities according to the longitude and latitude and generate other parameters, summarized in Table 4. We also use a synthetic dataset to test the scalability of our algorithms, shown in Table 5.

The algorithms are implemented in Visual C++ 2017, and the experiments are conducted in a Windows 10 machine with Intel(R) Core(TM) i5-6500 3.20GHz CPU and 16GB main memory. When the changes come, we treat all the events as new added and get a global plan as the contrast experiment, named Re-Plan. In each experiment, we repeat 10 times and report the average results.

4.2. Results on Real Dataset. In this section, we test the performance of our algorithms, denoted as B-De, η-De, $t^s - t^e$, and *Mul*, on Meetup datasets. For each algorithm, we randomly select several users to decrease their travel budget and several events to decrease their upper bound of participations one by one. For the multiple case, we randomly select some users and events of the three kinds of changes.

The experiment results are shown in Tables 6–9. The utility scores of B-De are the same as Re-Plan. The reason is that B-De conducts the same algorithm to update the plans of events as Re-Plan. In other cases, the algorithms are similar with Re-Plan. η-De is better than Re-Plan on Hawaii and Vancouver, $t^s - t^e$ is better on Hawaii and Hong Kong, and *Mul* is better on Singapore and Vancouver. In IBPSP problem, the stable plan is not unique; we can find different stable plans through different algorithms. Though the utility scores are not the same, the plans are still stable. In all the algorithms, the running time is much smaller than the Re-Plan; the reason is that the incremental algorithms only adjust the plans of users and events that are affected by the changes, but the Re-Plan makes plans for all the users and events over again. The running time increases as the data size increases. The incremental algorithms cost the similar memory as Re-Plan due to the fact that they all need memory to store the preference and the final plans. Mul costs a bit more memory than others due to size of U_{ac} and E_{ac} is larger than others.

4.3. Results on Synthetic Datasets. In this section, we test the scalability of the proposed algorithms on synthetic datasets. We first set $|U|$ as 5000 and change $|E|$ from 200 to 5000. Then, we set $|E|$ as 1000 and change $|U|$ from 200 to 5000.

The results of efficiency are shown in Figure 3. The total utility increases as the data size increases. The total utility of B-De is still as same as Re-Plan as shown in Figures 3(a) and 3(b). η-De is better than Re-Plan as $|E|$ increases. $t^s - t^e$ and Mul are better than Re-Plan as $|U|$ and $|E|$ increase.

The results of effectiveness are show in Figure 4. Both the running time and memory cost increase as $|E|$ and $|U|$ increase. $t^s - t^e$ and *Mul* cost more time than the others, the reason is that the plans adjusted in the two algorithms are more than the others. B-De runs faster than η-De when $|E|$ increases but runs the slowest when $|U|$ increases. The reason is that B-De needs to enumerate users while adjusting plans of events; the increasing of $|U|$ affects the running time. To the contrary, η-De needs to enumerate events while adjusting plans of users; the running time increases faster as $|E|$ increases.

5. Related Work

In this section, we review related works from two categories, event based social networks (EBSNs) and stable matching problem.

Studies on EBSNs. By analyzing Meetup and Plancust, X. Liu et al. [1] firstly proposed the concept of EBSN and received more and more attention [8–17]. On the event recommendation problem, Zhang et al. [9] predicted whether a user will participate in some events by learning the historical events that the user participated in. Cao et al. [18] scored users based on location information, attributes, and relations and built a Bayesian model to recommend events. Wang et al. [19] proposed a context-enhanced method to recommend events to users. However, the method relies too much on social information and geographical information; the lack and inaccuracy of data will seriously affect the accuracy. On the event organization problem, Shen et al. [3, 4] considered

Table 6: Results of B-De on real datasets.

City	Re-Plan			B-De		
	Utility	Time(s)	Memory(MB)	Utility	Time(s)	Memory(MB)
Beijing	218.94	0.81	0.78	218.94	0.01	0.78
Auckland	2443.73	4.62	2.71	2443.73	0.07	2.69
Hawaii	12469.67	56.19	166.58	12469.67	1.81	160.73
Hong Kong	18898.82	293.87	366.07	18898.82	6.35	370.36
Singapore	52753.51	11442.20	2902.35	52753.51	29.79	2900.78
Vancouver	70732.40	86492.48	12908.62	70732.40	103.67	13001.34

Table 7: Results of η-De on real datasets.

City	Re-Plan			η-De		
	Utility	Time(s)	Memory(MB)	Utility	Time(s)	Memory(MB)
Beijing	215.77	0.77	0.78	208.78	0.01	0.78
Auckland	2394.26	3.65	2.89	2244.57	0.08	2.75
Hawaii	12489.57	60.78	170.86	12604.53	2.35	177.75
Hong Kong	18796.87	301.78	359.17	18702.48	7.56	388.47
Singapore	52766.12	11304.18	2801.87	52689.48	26.88	2857.44
Vancouver	70655.31	86677.14	12944.63	70689.14	99.57	12903.78

Table 8: Results of $t^s - t^e$ on real datasets.

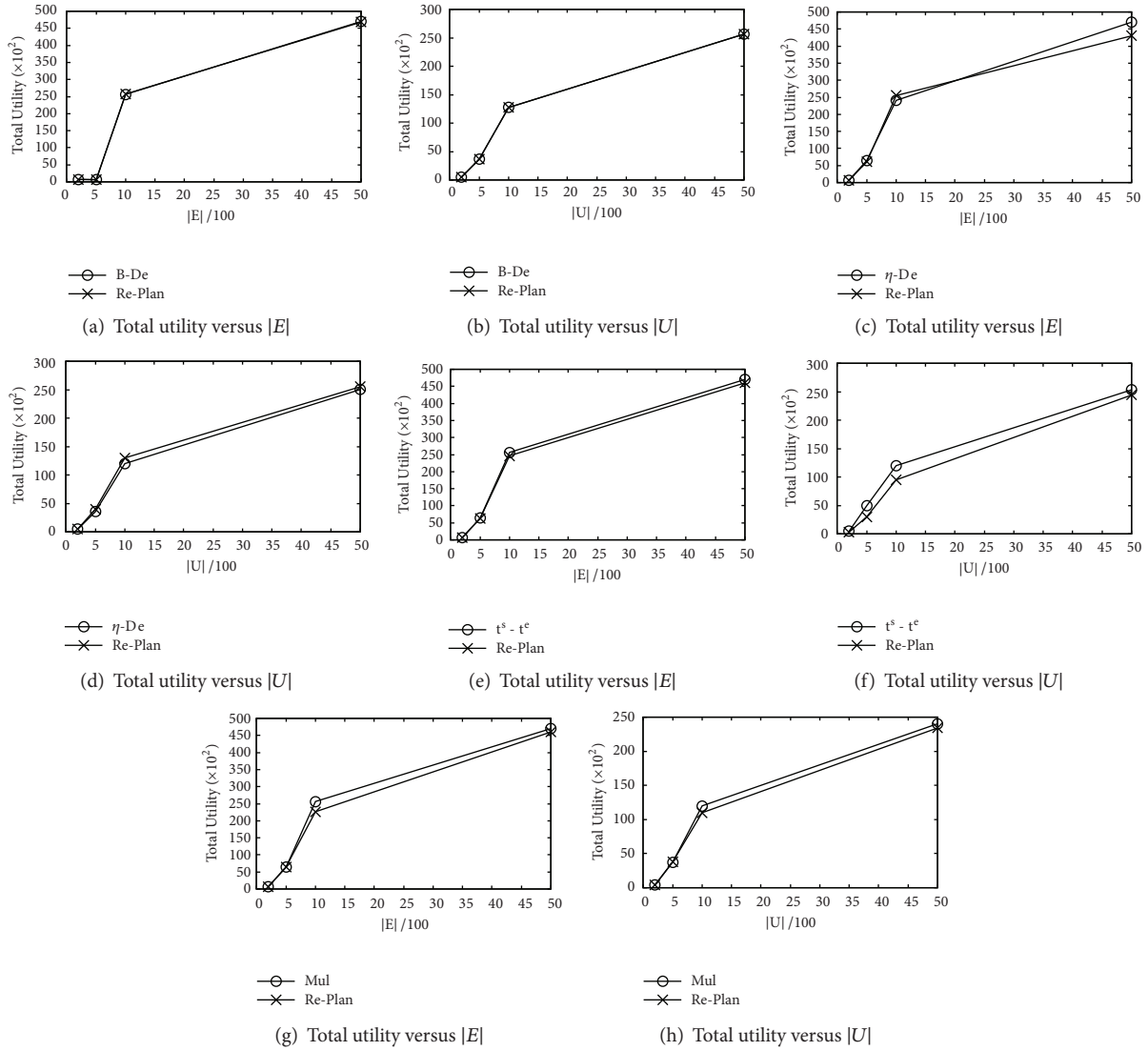
City	Re-Plan			$t^s - t^e$		
	Utility	Time(s)	Memory(MB)	Utility	Time(s)	Memory(MB)
Beijing	221.07	0.97	0.78	211.77	0.01	0.79
Auckland	2507.47	5.21	2.67	2487.45	0.08	3.04
Hawaii	11847.70	55.78	150.43	12078.8	2.02	184.48
Hong Kong	19752.41	321.47	397.15	19987.56	7.59	394.51
Singapore	52078.86	11648.67	2784.45	51978.07	33.48	2875.43
Vancouver	71097.98	87124.35	13478.74	70988.46	112.78	13998.47

Table 9: Results of *Mul* on real datasets.

City	Re-Plan			*Mul*		
	Utility	Time(s)	Memory(MB)	Utility	Time(s)	Memory(MB)
Beijing	202.17	1.21	0.81	198.45	0.01	0.78
Auckland	2378.47	5.14	2.07	2245.87	0.07	2.88
Hawaii	11987.78	60.74	152.78	11624.47	1.75	187.57
Hong Kong	18048.14	318.45	401.65	17956.24	5.33	407.98
Singapore	52417.64	11620.78	2788.79	52745.78	38.78	2998.47
Vancouver	70574.78	87144.54	13407.55	70741.34	121.52	14002.46

the conflict between events and made arrangements for users that avoid conflicts. Authors of [6] took into account the users' travel budget and extended the problem from simply matching the users with events to scheduling reasonable travels for users to participate in events. Cheng et al. [2] further considered the lower bound of the number of participants in the events and the issue of dynamic planning. The existing works only considered the preference of users, none of them studied on the bilateral preference stable planning problem. The authors of [20] studied a kind of stable matching problem over EBSNs; however, our work studies an incremental problem which is different from it.

Studies on Stable Matching Problem. The stable marriage (SM) problem was first proposed by Gale and Shapley [21] in 1962. Suppose a set of man and a set of woman with the same size; a stable marriage for these men and women is that everyone is matched with a partner and there are no such two couples



(a) Total utility versus $|E|$ (b) Total utility versus $|U|$ (c) Total utility versus $|E|$

(d) Total utility versus $|U|$ (e) Total utility versus $|E|$ (f) Total utility versus $|U|$

(g) Total utility versus $|E|$ (h) Total utility versus $|U|$

Figure 3: Efficiency on synthetic datasets.

that the man and the woman prefer each other than his/her partner. Reference [22] proved that the stable marriage always exists. There are some extensions of stable marriage problem, SM with incomplete preference lists, SM with preference lists with ties, and SM with incomplete preference lists with ties. In SM with incomplete preference lists, each person's preference list may be incomplete. Authors of [23] partitioned the set of men (women) into two sets: one is the set of men (women) who have partners in all stable matching, and the other is the set of men (women) who are single in all stable matching. In SM with preference lists with ties, one can include two or more persons with the same preference in a tie. Authors of [24] found a weakly stable matching in polynomial time. The last extension allows both incompleteness and ties in preference list. There series of approximation algorithm [25–28] and the approximation ratio reached 1.8 in [28]. The definition of stable matching problem is different from ours and the attributes are not changed dynamically; the existing studies cannot be applied.

6. Conclusions

In this paper, we define the Incremental Bilateral Preference Stable Planning (IBPSP) problem, which dynamically make plans for all the users to participate in suitable events. In this problem, we consider the upper bound of participants, event time conflicts, and the travel budget of users and dynamically update the plans when the constraints change. We propose several algorithms to update the plans when the changes come. We verify the effectiveness, efficiency, and scalability of the proposed methods through extensive experiments on both read and synthetic datasets.

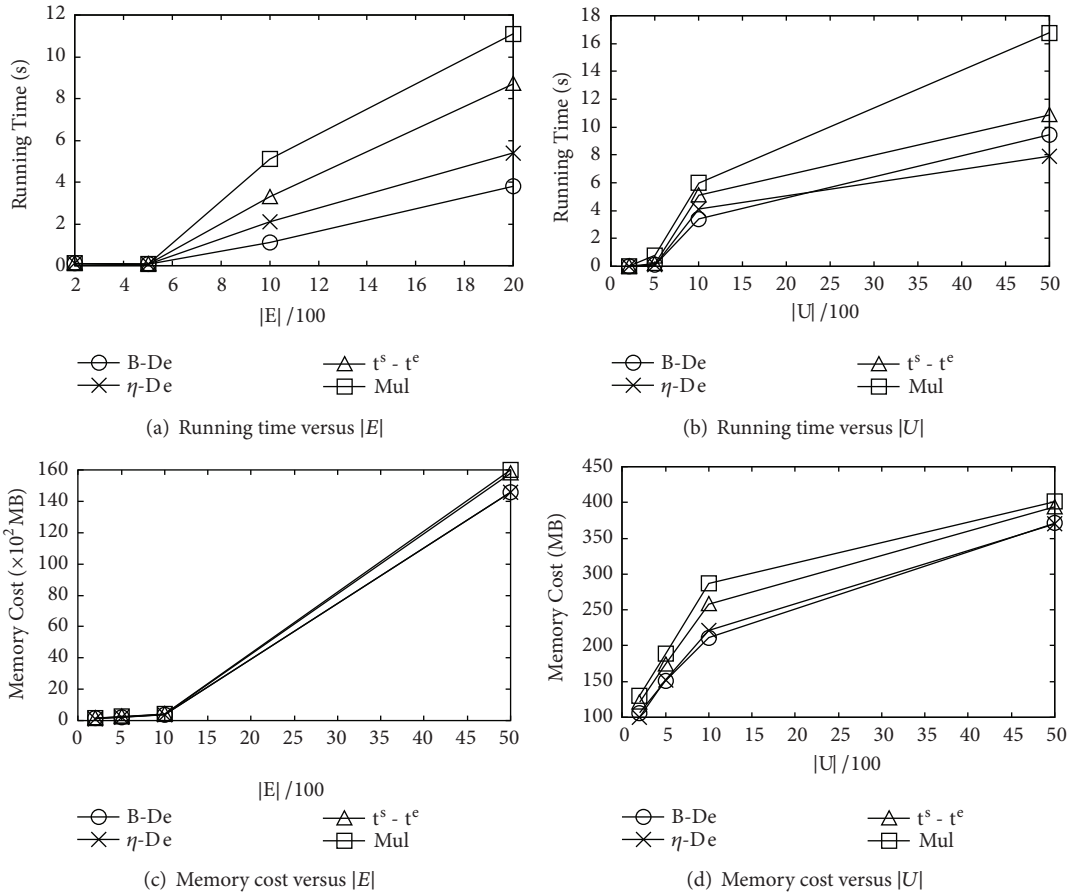


Figure 4: Effectiveness on synthetic datasets.

Data Availability

The Meetup data supporting the finding of this study are from previously reported studies and datasets, which have been cited as [1]. The processed data are available from the corresponding author upon request.

Conflicts of Interest

The authors declared that they have no conflicts of interest to this work

Acknowledgments

The work is supported by the National Key R&D Program of China (Grant no. 2016YFC1401900), the National Natural Science Foundation of China (Grants nos. 61332006, 61332014, 61328202, U1401256, 61572119, 61622202, 61572121, and 61702086), the Fundamental Research Funds for the Central Universities (Grants nos. N150402005, N171604007, and N171904007), the Natural Science Foundation of Liaoning Province (Grant no. 20170520164), and the China Postdoctoral Science Foundation (Grant no. 2018M631806).

References

[1] X. Liu, Q. He, Y. Tian, W. Lee, J. McPherson, and J. Han, "Event-based social networks: linking the online and offline social worlds," in *Proceedings of the 18th ACM SIGKDD International Conference*, pp. 1032–1040, Beijing, China, August 2012.

[2] Y. Cheng, Y. Yuan, L. Chen, C. Giraud-Carrier, and G. Wang, "Complex event-participant planning & its incremental variant," in *Proceedings of the 33rd IEEE International Conference on Data Engineering, ICDE 2017*, pp. 859–870, April 2017.

[3] J. She, Y. Tong, L. Chen, and C. C. Cao, "Conflict-aware event-participant arrangement," in *Proceedings of the 2015 IEEE 31st International Conference on Data Engineering (ICDE)*, pp. 735–746, Seoul, South Korea, April 2015.

[4] J. She, Y. Tong, L. Chen, and C. C. Cao, "Conflict-aware event-participant arrangement and its variant for online setting," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 9, pp. 2281–2295, 2016.

[5] K. Li, W. Lu, S. Bhagat, L. V. Lakshmanan, and C. Yu, "On social event organization," in *Proceedings of the 20th ACM SIGKDD international conference*, pp. 1206–1215, August 2014.

[6] J. She, Y. Tong, and L. Chen, "Utility-aware social event-participant planning," in *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2015*, pp. 1629–1643, June 2015.

[7] S. Qiao, N. Han, J. Zhou, R.-H. Li, C. Jin, and L. A. Gutierrez, "SocialMix: A familiarity-based and preference-aware location suggestion approach," *Engineering Applications of Artificial Intelligence*, vol. 68, pp. 192–204, 2018.

[8] K. Feng, G. Cong, S. S. Bhowmick, and S. Ma, "In search of influential event organizers in online social networks," in *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, SIGMOD 2014*, pp. 63–74, USA, June 2014.

[9] X. Zhang, J. Zhao, and G. Cao, "Who will attend? - predicting event attendance in event-based social network," in *Proceedings of the 16th IEEE International Conference on Mobile Data Management, MDM 2015*, pp. 74–83, USA, June 2015.

[10] W. Zhang, J. Wang, and W. Feng, "Combining latent factor model with location features for event-based group recommendation," in *Proceedings of the 19th ACM SIGKDD International Conference*, p. 910, August 2013.

[11] R. Du, Z. Yu, T. Mei, Z. Wang, Z. Wang, and B. Guo, "Predicting activity attendance in event-based social networks: Content, context and social influence," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2014*, pp. 425–434, 2014.

[12] T.-A. N. Pham, X. Li, G. Cong, and Z. Zhang, "A general graph-based model for recommendation in event-based social networks," in *Proceedings of the 2015 IEEE 31st International Conference on Data Engineering*, pp. 567–578, 2015.

[13] Xiaoyang Liu, Chao Liu, and Xiaoping Zeng, "Online Social Network Emergency Public Event Information Propagation and Nonlinear Mathematical Modeling," *Complexity*, vol. 2017, Article ID 5857372, 7 pages, 2017.

[14] W. Hu, H. Wang, C. Peng, H. Liang, and B. Du, "An event detection method for social networks based on link prediction," *Information Systems*, vol. 71, pp. 16–26, 2017.

[15] J. She, Y. Tong, L. Chen, and T. Song, "Feedback-Aware social event-participant arrangement," in *Proceedings of the 2017 ACM SIGMOD International Conference on Management of Data, SIGMOD 2017*, pp. 851–865, May 2017.

[16] Y. Tong, J. She, and R. Meng, "Bottleneck-aware arrangement over event-based social networks: the max-min approach," *World Wide Web*, vol. 19, no. 6, pp. 1151–1177, 2016.

[17] G. Li, Y. Liu, B. Ribeiro, and H. Ding, "On group popularity prediction in event-based social networks," in *Proceedings of the 12th International AAAI Conference on Web and Social Media*, pp. 644–647, June 2018.

[18] J. Cao, Z. Zhu, L. Shi, B. Liu, and Z. Ma, "Multi-feature based event recommendation in event-based social network," *International Journal of Computational Intelligence Systems*, vol. 11, no. 1, pp. 618–633, 2018.

[19] Z. Wang, P. He, L. Shou, K. Chen, S. Wu, and G. Chen, "Toward the new item problem: context-enhanced event recommendation in event-based social networks," in *Proceedings of the European Conference on Information Retrieval*, vol. 9022 of *Lecture Notes in Computer Science*, pp. 333–338, Springer International Publishing, 2015.

[20] Y. Cheng, G. Wang, B. Li, and Y. Yuan, "Bilateral preference stable planning over event based social networks," *Journal of Software*, vol. 30, pp. 573–588, 2019.

[21] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *The American Mathematical Monthly*, vol. 69, no. 1, pp. 9–15, 1962.

[22] A. E. Roth, "Deferred acceptance algorithms: history, theory, practice, and open questions," *International Journal of Game Theory*, vol. 36, no. 3-4, pp. 537–569, 2008.

[23] D. Gale and M. Sotomayor, "Some remarks on the stable matching problem," *Discrete Applied Mathematics: The Journal of Combinatorial Algorithms, Informatics and Computational Sciences*, vol. 11, no. 3, pp. 223–232, 1985.

[24] R. W. Irving, "Stable marriage and indifference," *Discrete Applied Mathematics: The Journal of Combinatorial Algorithms, Informatics and Computational Sciences*, vol. 48, no. 3, pp. 261–272, 1994.

[25] K. Iwama, S. Miyazaki, and K. Okamoto, "A (2-clog N/N)-approximation algorithm for the stable marriage problem," in *Proceedings of the 2004 Scandinavian Workshop on Algorithm Theory*, vol. 3111 of *Lecture Notes in Computer Science*, pp. 349–361, Springer, Berlin, 2004.

[26] K. Iwama, S. Miyazaki, and N. Yamauchi, "A 2-c$(1/\sqrt{n})$)-approximation algorithm for the stable marriage problem," in *Proceedings of the 2005 International Symposium on Algorithms and Computation*, vol. 3827 of *Lecture Notes in Computer Science*, pp. 902–914, Springer, Berlin, 2005.

[27] K. Iwama, S. Miyazaki, and N. Yamauchi, "A 1.875-approximation algorithm for the stable marriage problem," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007*, pp. 288–297, ACM, New Orleans, La, USA.

[28] S. Khuller, "Problems column," *ACM Transactions on Algorithms*, vol. 3, no. 3, 2007.

Hindawi
Complexity
Volume 2019, Article ID 4932030, 16 pages
https://doi.org/10.1155/2019/4932030

Research Article

Predicting Quality of Service via Leveraging Location Information

Liang Chen,[1] **Fenfang Xie**,[1] **Zibin Zheng**,[1,2] **and Yaoming Wu**[1]

[1]*School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China*
[2]*National Engineering Research Center of Digital Life, Sun Yat-sen University, Guangzhou 510006, China*

Correspondence should be addressed to Zibin Zheng; zhzibin@mail.sysu.edu.cn

Received 25 January 2019; Accepted 26 March 2019; Published 11 April 2019

Guest Editor: Jianxin Li

QoS (Quality of Service) (our approach can be applied to a wide variety of services; in this paper, we focus on Web services) performance is intensively relevant to locations due to the network distance and the Internet connection between users and services. Thus, considering the location information of services and users is necessary. However, the location information has been ignored by most previous work. In this paper, we take both services' and users' location information into account. Specifically, we propose a location-aware QoS prediction approach, called LANFM, by exploiting neural network techniques and factorization machine to improve user-perceived experience. First of all, the information (e.g., id and location) of services and users is expressed as embedding vectors by leveraging neural network techniques. Then, the inner product of various embedding vectors, along with the weighted sum of feature vectors, is used to predict the QoS values. It should be noted that the inner product operation could capture the interactions between services and users, which is helpful to predict QoS values of services that have not been invoked by users. A collection of extensive experiments have been carried out on a real-world dataset to validate the effectiveness of the LANFM model.

1. Introduction

Web services are service-oriented architecture technologies that provide services through standard Web protocols to support the interactive operation of different machines on the network [1]. With the exponential growth of Web services on the Internet, there emerge lots of Web services that have identical or similar functionalities. These Web services have different QoS performance. The nonfunctional attributes of Web services are extensively depicted by QoS [2]. Generally speaking, QoS is a list of nonfunctional performance criteria of Web services, including popularity, response time, throughput, failure probability, and availability. The user-perceived QoS associates closely with network status, geographical location, and service runtime environment.

In reality, only a part of Web services have been invoked by users. As a result, the invocation matrix between Web services and users is considerably sparse, since most of the entries in the user-service matrix are null. Evaluating Web service is a good way to get accurate QoS for service users. However, there are a lot of challenges in invoking the Web services for evaluation purpose from the client user's perspective. Firstly, it is time-consuming and impractical for users to invoke every single Web service for evaluation purpose, due to the massive Web services on the Internet. Secondly, most Web service providers are commercial companies. These companies allow users to acquire QoS information by invoking Web services, while they can be prohibitively expensive for users. Thirdly, it is not professional enough for inexperienced users to conduct evaluation on Web services. Lastly, conducting service invocations periodically brings a heavy workload to users to observe the services' QoS performance constantly. Therefore, how to accurately predict the QoS values is becoming an urgent problem to enhance the user-perceived experience.

To alleviate these critical challenges, much work [3–6] has been done on QoS prediction, while, in most cases, only the information of user id and service id is utilized and the location information of users and services is largely ignored. It should be noted that QoS performance of Web

services is inextricably linked with locations. The reason is that the network distance and the Internet connections between users and services have impact on user-perceived QoS performance to a large extent. Thus, taking location factors into account is useful to improve the QoS prediction results. Recently, a few works [7–12] have noticed the effect of users' location on QoS values. These investigations are mainly based on the observation that when invoking the same Web service, users in different places may go through distinct experience due to the diverse physical infrastructure. However, these existing QoS prediction methods have the drawbacks of high dimension, high time complexity, and high expense, due to utilizing traditional user-based/item-based collaborative filtering and matrix factorization techniques. The dimensions of the feature vectors used in previous work are relatively high, with a time complexity basically in $O(N^2)$, and it requires a lot of artificial feature engineering. More innovative and effective methods are required to employ the service information and the user information. These methods promote solving the above drawbacks and further enhancing the accuracy of QoS prediction.

To overcome the above-mentioned drawbacks of existing QoS prediction methods, we prefer to exploit the neural network and factorization machine technique. Specifically, the time complexity of factorization machine is linear. In addition, factorization machine only relies on a linear number of parameters [13]. Embedding techniques in neural networks can project data from high-order source space to low-order target space and can maintain good structural invariance. The problem of high dimension can be well solved by embedding technique. The interactions between Web services and users can be captured by factorization machine via learning cross features, even if the Web services have never been invoked by the users. Factorization machine can be applied to dispose the drawbacks of data sparsity and high time complexity.

Based on the characteristics of our QoS prediction problem and the significance of embedding and factorization machine, in this paper, we propose a *L*ocation-*A*ware QoS prediction approach, by exploiting *N*eural network and *F*actorization *M*achine (called LANFM), to improve the user-perceived experience. This paper is extended from its preliminary conference version [14], which mainly takes advantage of users' id feature and services' id feature for QoS prediction. Moreover, we only consider impact parameters like matrix density and dimension of embedding vector (matrix density is set as 10%) in the conference paper.

Particularly, the significant contributions of this paper could be summed up as three-fold:

(1) Both services' and users' location information is taken into consideration. The embedded form of services' and users' location information can be used to capture the interactions between services and users

(2) Based on the additional location information, the interactions between users and services can be enhanced. Namely, the proposed approach is an enhanced neural factorization machine model for QoS prediction

(3) A real-world dataset is used to validate the effectiveness of our LANFM model. Empirical study shows that considering users' and services' location information is indeed helpful to improve the accuracy of QoS prediction. Impacts of the matrix density, the dimension of the embedding vector, and the batch size are evaluated, respectively

The remaining part of the paper is arranged as follows. Section 2 highlights some related work on collaborative filtering and location-aware QoS prediction. Section 3 describes the QoS prediction problem, motivation, and the framework of our LANFM model. Section 4 presents the details of our LANFM model for predicting QoS values. Section 5 introduces some empirical studies and analyzes the experimental results. Section 6 makes a conclusion of the paper.

2. Related Work and Discussion

In this section, we present some investigations associated with the research problem of QoS prediction.

2.1. Collaborative Filtering Based QoS Prediction. Currently, collaborative filtering technique is widely applied to QoS prediction. These collaborative filtering based methods are mainly classified into three categories: memory-based, model-based, and hybrid. We will introduce them individually in the following paragraphs.

Memory-Based Approach. This kind of approach measures the similarity between users or services by exploiting the historical invocation logs between users and services. It involves user-based approaches [15, 16], item-based approaches [17, 18], and the hybrid of them [4, 19]. For instance, Sun et al. [20] presented a novel method to measure the similarity between Web services. Further, they proposed to predict QoS by introducing normal recovery collaborative filtering. Xiong et al. [21] exploited the historical usage experience and proposed a collaborative approach. They aimed to address the QoS prediction problem on unbalanced data. Ma et al. [22] mined several significant characteristics on some QoS datasets that had never been found before. They put forward a powerful prediction method to realize these characteristics to help predict QoS.

Model-Based Approach. This type of approach predicts QoS for service users by utilizing machine learning techniques. Some representative approaches of this type are presented in the following. Zheng et al. [5] proposed a neighborhood integrated matrix factorization approach to predict QoS for users by considering the historical invocation records between users and services. Xu et al. [23] took the users' reputation into consideration and presented a reputation-based matrix factorization approach to predict QoS values. Luo et al. [24] proposed a matrix-factorization model with Tikhonov regularization terms and under a nonnegativity constraint for QoS prediction. Wu et al. [14] embedded

the user id and the service id to vectors and employed factorization machine to predict QoS for users.

Hybrid Approach. This style of approach integrates memory-based and model-based approaches. The hybrid methods usually unify the power of similarity measurement and matrix factorization. For example, Chen et al. [25] assembled the neighborhood relations of users and services. Then, they predicted QoS values by applying a neighborhood regularized matrix factorization approach. Su et al. [6] firstly combined the direct similarity and the transitive indirect similarity of services. Further, they proposed a hybrid algorithm to perform QoS prediction by integrating the nonnegative matrix factorization model and the expectation-maximization. Lo et al. [26] identified neighborhoods via measuring the similarities on the user side and the service side from different aspects. Then, they proposed a relational regularized matrix factorization structure for QoS prediction.

2.2. Location-Aware QoS Prediction. Users' and Web services' location information is intensively relevant to QoS performance of Web services because of the network distance and Internet connections. Considering location information will do a great favor to enhance the QoS prediction accuracy.

Some studies have taken the location information into consideration recently. Chen et al. [27] employed the characteristics of QoS by designing a region model for large-scale services' QoS prediction. Lo et al. [28] identified neighbors by considering the local connectivity and geographical information. They proposed a novel location-based regularized matrix factorization model to predict QoS for users. He et al. [29] applied the location information of clustered user-service groups and put forward a hierarchical matrix factorization method based on the location information to achieve personalized QoS prediction. Chen et al. [11] divided users and services into several groups by employing users' location information and services' location information, along with the QoS values. They proposed an innovative collaborative filtering model to select services with the best QoS performance, for users. Liu et al. [9] presented a memory-based method for service QoS prediction by leveraging both users' and services' location information. The location information can help choose similar neighbors for the objective user or the objective service. Wu et al. [7] used the invocation record between users and services, and a generalized context-sensitive matrix factorization method was proposed to predict QoS values for services. Kuang et al. [8] took advantage of the user's reputation and the user's and the service's location information, and a personalized QoS prediction method was proposed to solve the problem of data sparsity, cold start, and data incredibility.

All the studies mentioned above are useful for predicting QoS value for users to a certain extent. But most of the above methods suffer from high dimension, high time complexity, and high expense. In the proposed LANFM model, we employ the embedding feature extracting techniques to represent the implicit vectors of services' and users' location information. By applying embedding technique, we can resolve the drawback of high dimension. In addition, embedding vectors can be utilized to mine the potential correlations between services and users, even if the user has not invoked the service yet. Moreover, the proposed LANFM model could be calculated in linear time; thus it can solve the problem of high time complexity.

Users \ Services	S1	S2	S3	S4	S5
U1	0.175	?	0.479	1.644	?
U2	?	0.177	4.909	?	0.355
U3	6.413	?	7.813	5.338	?
U4	5.704	5.381	?	?	9.418

QoS Prediction

Users \ Services	S1	S2	S3	S4	S5
U1	0.175	5.532	0.479	1.644	12.514
U2	2.366	0.177	4.909	5.459	0.355
U3	6.413	3.361	7.813	5.338	15.331
U4	5.704	5.381	5.358	6.059	9.418

Figure 1: An example of QoS prediction.

3. Motivation and Framework

In this part, we define the problem to be solved in Section 3.1. We present the motivation of incorporating location information in Section 3.2. We introduce the framework of the proposed LANFM model in Section 3.3.

3.1. Problem Description. We aim to achieve accurate QoS prediction for users by making full use of the historical QoS invocation records, services' location information, and users' location information in this paper. The details of the problem are described as follows.

Suppose $U = \{u_1, u_2, \ldots, u_m\}$ and $S = \{s_1, s_2, \ldots, s_n\}$ are a set of m users and n services, respectively. $Q \in R^{m \times n}$ is the matrix between m users and n services. q_{ij} is the element in Q, and many of their values are missing. Our goal is to predict the missing values in the matrix Q by exploiting the relationships between the existing elements.

An example is displayed in Figure 1 to help comprehend the problem of QoS prediction. The invocation matrix between users (e.g., U1, U2, U3, U4) and services (e.g., S1, S2, S3, S4, S5) is exhibited in the upper part of Figure 1. Every element in the invocation matrix stands for a QoS attribute value (e.g., response time or throughput). Then, the problem we investigate is transformed to how to accurately predict the missing elements in the invocation matrix based on the existing elements. It could be found that only a part

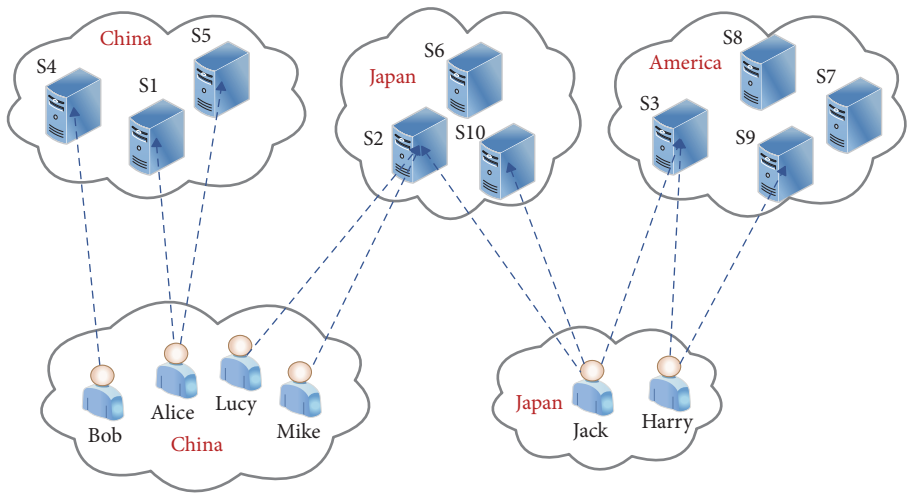


Figure 2: A toy example of the influence of the location information.

of services have been invoked by users. For instance, U1 has invoked S1, S3, and S4. U4 has observed QoS information on S1, S2, and S5. There still exist a small amount of common services, although not every user has invoked all services. We can complete the invocation matrix by applying the idea of collaborate filtering with the help of these common services. The complete matrix is shown in the lower part.

3.2. Motivation. The user-perceived QoS (such as throughput, reliability, and response time) associates with network status, geographical location, and service runtime environment closely. Network performance is related to network bandwidth, network latency, and network distance intensively. Among the several network performance influence factors, the location information is one of the key factors. Location information can affect the behavior of users [30]. In general, if the Web service is deployed in the user's network (e.g., a local area network or a subnet), then, the Web server is more likely to respond to the user in a short period of time. That is, the response time between the user and the Web server is short, when a user invokes the Web service. Conversely, if the Web service is deployed far away from the user (e.g., across multiple subnets), there will be a long time before the Web server responds to the service user. This phenomenon is the same with the fact that when a user visits local websites, the Web page will soon respond to him/her. When the user visits foreign websites, it is often relatively slow to respond to him/her. This is because long-distance requests and responses on the Internet often need to be routed and forwarded multiple times, which is time-consuming. Figure 2 shows a toy example to illustrate why considering services' and users' location information to predict services' QoS is important.

Assuming that Alice and Jack are two different service users. They lie in two remote networks. Web servers (e.g., S1, S2, and S3) are deployed in different kinds of networks and provide similar services. When Alice requests services, server S1 will answer the requirement of Alice and provide services to her, since the network distance between Alice and S1 is closer than that of S2 and S3. However, if S1 is invalid or has high latency, it will not answer Alice in time. At the same time, Alice is in a hurry to ask for a certain service. Therefore, Alice would like to send a request to S2 and S3. She intends to send a request to S2, since the network distance between Alice and S2 is closer than that of S3. But Alice never requests services from S2; she is not sure whether S2 can be used. Thus, it is essential to predict the QoS on server S2 such that it can provide services to Alice in the condition that S1 is out-of-service and avoids wasting time. Based on the idea of collaborative filtering, if Lucy or other neighbors have requested a service on server S2 before, she/they can do a great favor for predicting the QoS on server S2. By considering users' and services' location information and predicting the QoS of services, we can select appropriate services for users and improve the user-perceived experience.

3.3. Framework of LANFM Model. This section introduces the framework of the LANFM model for predicting QoS values in Figure 3. The detailed procedures are given in the following:

(i) The information of users and services is firstly collected to be stored in the local database. After a series of data processing, users' location information, services' location information, user-service invocation matrix (e.g., the historical QoS information that users perceive on services), and other additional requisite information can be obtained

(ii) The user information (e.g., user id, user ASN (autonomous system number) id, and user country id) and the service information (e.g., service id, service ASN id, and service country id) are expressed as one-hot encoding vectors (it is a way of encoding; for an n-dimensional vector, only one element is 1, and the others are 0. The details will be introduced in Section 4). Further, these one-hot encoding vectors

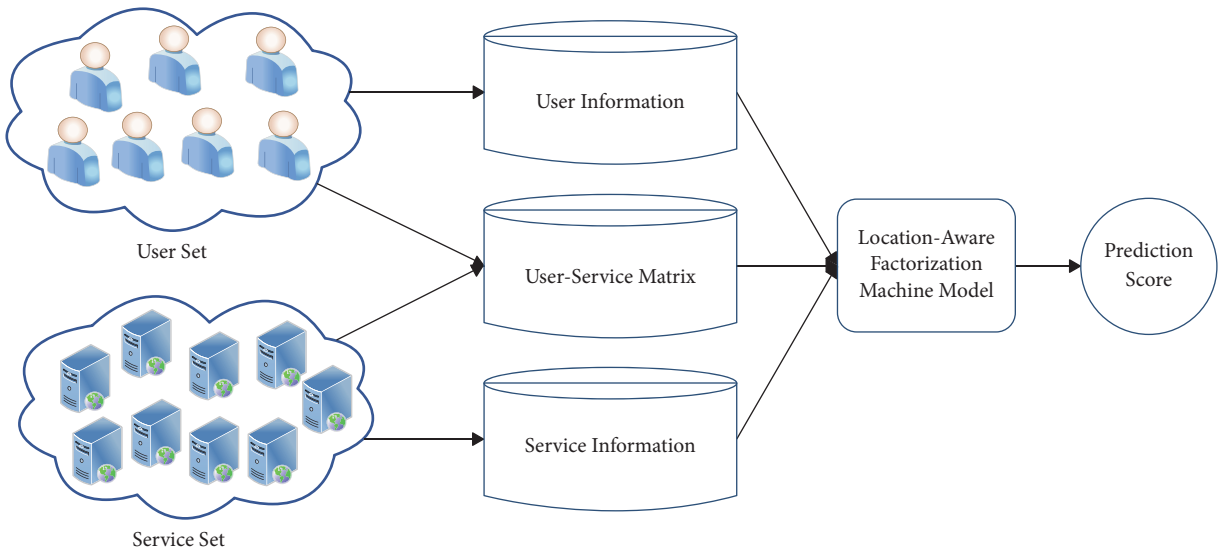


Figure 3: Framework of our LANFM model.

are concatenated as an input feature vector for every historical usage record

(iii) The one-hot encoding vectors are represented as embedding vectors by exploiting neural network techniques. Then, the inner products between users' embedding vectors and services' embedding vectors are used to capture the interactions between users and services

(iv) The QoS value for pairwise user-service can be predicted by summing the inner product of embedding vectors and the weighted sum of feature vectors, namely, the embedding based location-aware factorization machine model

Note that an AS (autonomous system) is a small unit that has the authority to autonomously determine which routing protocol to use in the system. AS is an independently manageable network unit (e.g., a university, an enterprise, or a company). Every AS has an individual id. It is known as ASN. ASNs are important because the ASN uniquely identifies each network on the Internet.

Figure 3 gives the overall framework of our LANFM model. The core part of our approach is the location-aware factorization machine model. To better describe the model, we give an illustration as shown in Figure 4. Firstly, the user information (user id, user ASN id, and user country id) and the service information (service id, service ASN id, and service country id) are expressed as one-hot encodings (e.g., one-hot encoding vectors: (1 0), (0 1 0), (1 0 0 0 0), (0 1 0), (0 1 0 0), (1 0 0 0)). Secondly, the one-hot encoding forms of the user information and the service information are represented as various embedding vectors (e.g., $V_1, V_4, V_6, V_{12}, V_{15}, V_{18}$) by applying neural network technique. Thirdly, the inner products of embedding vectors (i.e., the user information embedding vectors and the service information embedding vectors (e.g., $\langle V_1, V_4\rangle, \langle V_4, V_6\rangle, \langle V_6, V_{12}\rangle, \langle V_{12}, V_{15}\rangle, \langle V_{15}, V_{18}\rangle$)), along with the weighted sum of feature vectors (e.g., $w_1x_1, w_4x_4, w_6x_6, w_{12}x_{12}, w_{18}x_{18}$), are leveraged to predict the QoS values. Finally, the prediction score for pairwise user-service can be acquired (e.g., $\widehat{y}$).

4. Location-Aware Factorization Machine Model

In this section, we give the details of our location-aware factorization machine approach for predicting QoS values. Our LANFM model mainly consists of three components: location information processing, embedding based factorization machine model, and model learning. The details of these components are shown in the following subsections.

4.1. Location Information Processing. The processing of Web services' and users' location information is presented in this section, which is the basis of our location-aware factorization machine model.

The original location information of Web services and users in our dataset is presented in Tables 1 and 2, respectively. Web services' location information includes Web service id, WSDL address (URL), service provider name, and service's country name. Users' location information consists of user id, user's IP address, user's country name, longitude, and latitude. To obtain the more exact services' location information and users' location information, we map the WSDL address and the user's IP address to ASN (Since the WSDL address of Web service is already known, it is easy to transfer the DNS (Domain Name System) of the URL to IP address and then map the IP address to ASN). The mapping operation is done by leveraging the GeoLite Autonomous System Number Database (http://www.maxmind.com) . After a series of processing, the final representations of Web services' location information and users' location information are correspondingly listed in Tables 3 and 4.

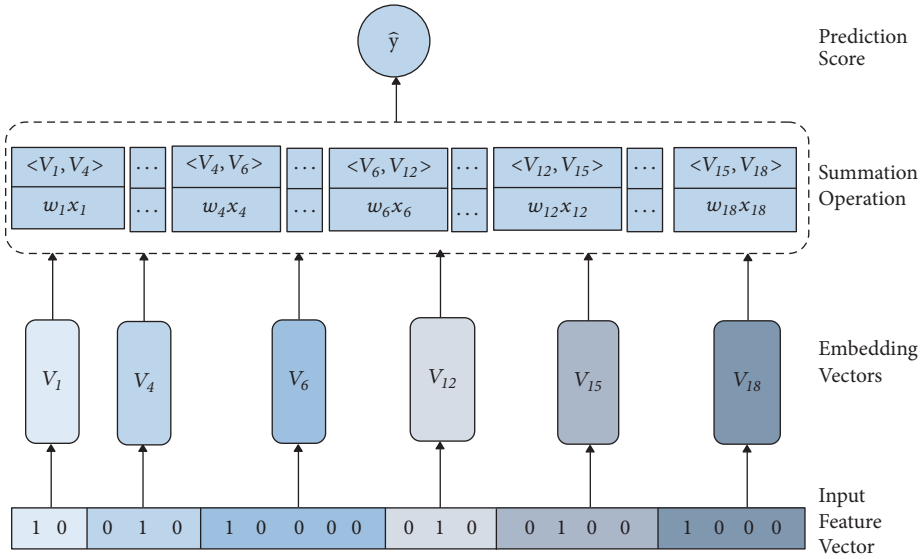


Figure 4: An illustration of LANFM model.

Table 1: Original location information of Web service.

Web Service ID	WSDL Address	Provider Name	Country Name
1	http://www.clearsale.com.br/aplicacao/entrada.asmx?WSDL	clearsale.com.br	United States
2	http://www.law.uni-sofia.bg/_vti_bin/People.asmx?wsdl	uni-sofia.bg	Bulgaria
3	http://www.etfo.ca/_vti_bin/Authentication.asmx?wsdl	etfo.ca	Canada
4	http://www.webxml.com.cn/WebServices/WeatherWebService.asmx?WSDL	webxml.com.cn	China
5	http://www.emris.cz/_vti_bin/BusinessDataCatalog.asmx?wsdl	emris.cz	Czech Republic

Table 2: Original location information of users.

User ID	IP Address of the User	Country	Longitude	Latitude
1	12.108.127.138	America	40.44	-79.96
2	122.1.115.91	Japan	36	138
3	128.233.252.11	Canada	52.13	-106.67
4	129.242.19.196	Norway	69.67	18.97
5	202.38.99.68	China	39.93	116.39

Table 3: Processed location information of Web services.

Web Service ID	Country Name	ASN
1	United States	33070
2	Bulgaria	5421
3	Canada	36031
4	China	23650
5	Czech Republic	43541

Note that we utilize the ASN instead of IP address or WSDL address since close IP addresses (e.g., 4.67.68.0 and 4.67.64.0) do not necessarily belong to the same AS or country (e.g., Canada and Japan). This demonstrates that employing IP address to represent Web services' and users' location information is probably not enough to recognize neighbor users or neighbor Web services. Moreover, the distance between users on the Internet is usually measured by utilizing the Internet AS-level topology [31]. The above analyses give a reason for adopting AS to represent a user's or service's location rather than other geographic positions. By representing users' and services' location information as the aforementioned form, it is accurate and easy for us to measure the closeness between users and Web services.

Table 4: Processed location information of users.

User ID	Country Name	ASN
1	America	7018
2	Japan	4713
3	Canada	22950
4	Norway	224
5	China	4538

4.2. Embedding Based Factorization Machine Model. We obtain the user id, user's country name, user's ASN, service id, service's country name, and service's ASN through the location information processing. Then, we transform user's country name and ASN to user's country id and ASN id, respectively. The service's country name and ASN also undergo the same transformation. Further, we utilize one-hot encoding to represent user id, service id, user's country id, user's ASN id, service's country id, and service's ASN id (i.e., the ID layer in Figure 5). One-hot encoding is an effective

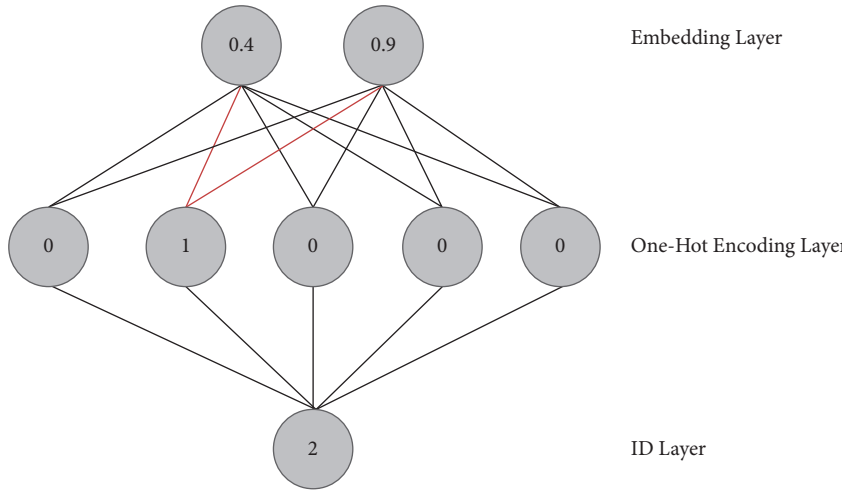


Figure 5: The procedure of embedding vectors.

Table 5: One-hot encoding representation of seven continents.

Name	One-hot Encoding
Asia	1000000
Africa	0100000
North America	0010000
South America	0001000
Antarctica	0000100
Europe	0000010
Oceania	0000001

encoding method, which uses n bit state registers to encode n states. Each state has its own register bit, and at any time, only one of them is valid [32]. Table 5 presents an example to express seven continents by one-hot encoding.

The mapping of one-hot encoding to embedding is through a fully connected layer. Embedding is a very popular neural network technique in recent years. It attempts to learn feature interactions from original input data. Embedding projects the data from high-order resource space to low-order target space [33]. The LANFM model can generalize to unobserved feature combinations and preserve the structure by leveraging embedding techniques. For example, we have two types of features: goods = {watch, necklace} with gender = {male, female}, and get a new cross features set: goods_gender = {watch-male, necklace-female, necklace-male, watch-female}. The cross features (also known as combination features) in the set mean that a male (or female) buys a watch (or a necklace). Assuming that we obtain a subset which collects the information that a male buys a watch, a female buys a watch and a female buys a necklace. But we do not get the information that a male buys a necklace and need to predict the possibility of this. It is hard to deal with this kind of problem through aforementioned methods. However, our LANFM model can handle the problem by learning the distributed representation of cross features and capturing the interactions between goods and gender.

The details of the mapping procedure are the following: first of all, the one-hot encoding representation of the user information and the service information is treated as the input feature of the fully connected layer (i.e., the one-hot encoding layer in Figure 5). Next, the weight of each link in the fully connected layer is calculated. Finally, the embedding representation of the user information and the service information is obtained through the fully connected layer (i.e., the embedding layer in Figure 5). The problem of high dimension can be tackled by embedding techniques. Note that the red lines in Figure 5 represent each of the dimensional values of the embedding vector. This procedure is a part of the location-aware factorization model, and the final embedding vectors are learned by stochastic gradient descent during the training process. Details about the model learning will be introduced in the following subsection.

Let X denote the input feature vector. It is concatenated by the one-hot encoding representation of the user information (e.g., user id, user's ASN id, and user's country id) and the service information (e.g., service id, service's ASN id, and service's country id). Then, the embedding based factorization machine model equation is defined as

$$\widehat{y}(X) = w_0 + \sum_{i=1}^{f} w_i x_i + \sum_{i=1}^{f-1} \sum_{j=i+1}^{f} \left\langle V_i, V_j \right\rangle x_i x_j, \quad (1)$$

wherein f is the length of the feature vector. V_i is the ith embedding vector. $\langle V_i, V_j \rangle$ is the inner product of various embedding vectors (e.g., user id embedding vector, user's country embedding vector, user's AS embedding vector, service id embedding vector, service's country embedding vector, and service's AS embedding vector). w_0 is the global bias, and w_i is the weight of the ith variable. The first two terms in the formula express the linear regression model. The third term is the 2-way cross feature which considers the association information between any two different features.

The formula of cross features is as follows:

$$\left\langle V_i, V_j \right\rangle = \sum_{l=1}^{k} v_{il} \cdot v_{jl}, \tag{2}$$

wherein k is a hyperparameter that determines the dimension of the embedding vectors. v_{il} is the lth value in the ith embedding vector of the user information or the service information.

The interactions between various embedding vectors can be reformulated as follows:

$$\begin{aligned} &\sum_{i=1}^{f-1} \sum_{j=i+1}^{f} \left\langle V_i, V_j \right\rangle x_i x_j \\ &= \frac{1}{2} \sum_{l=1}^{k} \left(\left(\sum_{i=1}^{f} v_{il} x_i \right)^2 - \sum_{i=1}^{n} v_{il}^2 x_i^2 \right). \end{aligned} \tag{3}$$

The derivation details of Eq. (3) are in [13].

4.3. Model Learning. To estimate the performance of our LANFM model, we conduct a loss function to evaluate the error between the estimated value and the real value. Mathematically, the loss function of the LANFM model is represented as

$$\min_{\theta} L\left(y, \hat{y}\right) = \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{n} I_{ij} \left(y_{ij} - \hat{y}_{ij}\right)^2, \tag{4}$$

where I_{ij} is the indicator function. If a user u_i uses a service v_j, I_{ij} is equal to 1; otherwise, it is equal to 0. Our goal is to minimize the sum of the squared errors. In this way, we can achieve the optimal loss.

Stochastic gradient descent (SGD) [34] is a universal optimization method in machine learning and deep learning. It randomly extracts a sample from the sample set at a time and updates it by gradient once after training. In the case of a large sample size, a model with an acceptable loss value can be obtained without training all the samples. In addition, the final result of the loss function is often in the vicinity of the global optimal solution. The optimization of the loss function given by Eq. (4) can be worked out by executing SGD in θ:

$$\frac{\partial L\left(y, \hat{y}\right)}{\partial \theta} = \left(\hat{y} - y\right) \frac{\partial \hat{y}}{\partial \theta}, \tag{5}$$

$$\frac{\partial \hat{y}}{\partial \theta} = \begin{cases} 1, & if\ \theta = w_0 \\ x_i, & if\ \theta = w_i \\ x_i \sum_{j=1}^{f} v_{jl} x_j - v_{il} x_i^2, & if \theta = v_{il} \end{cases} \tag{6}$$

The following formula is used to update the model parameters:

$$\theta = \theta - \eta \cdot \left(\hat{y} - y\right) \frac{\partial \hat{y}}{\partial \theta} \tag{7}$$

where θ is the model parameters (e.g., w_i, v_{il}), $\eta > 0$ is the learning rate, and η controls the speed of gradient descent.

4.4. Complexity Analysis. The main computation of our location-aware factorization machine model is the evaluation of the loss function L and its gradients against the variables. We only need to calculate all of the l in the formula $\sum_{j=1}^{f} v_{jl} x_j$ in the first time when iterating parameters, because $x_i \sum_{j=1}^{f} v_{jl} x_j - v_{il} x_i^2$ associates with l closely. And then, all gradients of v_{jl} can be obtained easily. Evidently, the complexity of calculating entire l in the formula $\sum_{j=1}^{f} v_{jl} x_j$ is $O(kf)$. The complexity of calculating every gradient of the parameter is $O(1)$ when $\sum_{j=1}^{f} v_{jl} x_j$ is known. After obtaining the gradient of parameters, the time complexity of updating parameters is $O(1)$. In total, there are $kf + f + 1$ parameters to be estimated in the LANFM model. Therefore, the time complexity of our LANFM model is $O(kf)$. In a word, the training time complexity of our LANFM model is linear. The above analysis process indicates the efficiency and scalability of our LANFM model.

5. Empirical Study

In this section, we carry out a series of empirical studies on a public dataset. Moreover, the LANFM model is compared against several baseline approaches to verify the effectiveness of our LANFM model, and we analyze the experimental results.

In the following subsections, the detailed statistics of the WSDream dataset is depicted in Section 5.1. The evaluation metrics are shown in Section 5.2. The performance comparison between the proposed LANFM model and other baseline methods is introduced in Section 5.3. The impact of parameters (e.g., matrix density, dimension of embedding vectors, and batch size [35]) (it is the size of each batch of data, i.e., how many samples are trained in each iteration) and experimental results discussion are presented in Sections 5.4, 5.5, and 5.6, respectively.

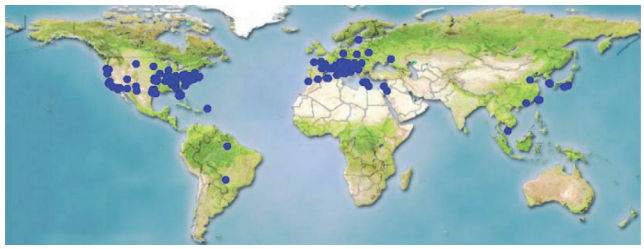
5.1. Dataset Description. We implement a collection of experiments on a real-world dataset: WSDream (https://wsdream.github.io/dataset/wsdream_dataset1.html). The location distribution of users and services is shown in Figure 6. The WSDream dataset includes 339 distributed users, 5,825 services, and 1,974,675 historical invocation logs between users and services. The numbers of users' ASs and Web services' ASs are 137 and 1021. The QoS properties in the WSDream dataset are response time and throughput. The values of response time range from 0 to 20. The throughput values vary from 0 to 1000. The details of our dataset are described in Table 6.

In this paper, we attempt to predict the above-mentioned two QoS attributes for users. Our LANFM model is suitable for predicting any QoS attributes, with proper modification of the QoS attribute in the user-service invocation matrix.

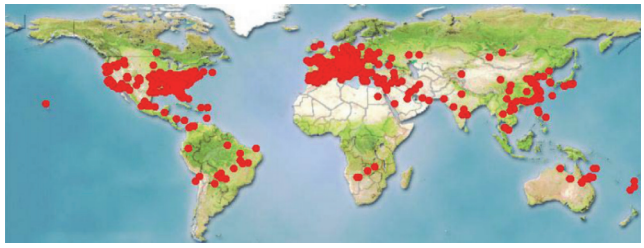
5.2. Evaluation Metrics. We employ two evaluation metrics, Mean Absolute Error (MAE) and Normalized Mean Absolute Error (NMAE), to measure the performance of our proposed

Table 6: Statistics of Web service QoS dataset.

Statistics	Values
Number of Service Users	339
Number of Web Services	5825
Number of Web Services Invocations	1,974,675
Range of Response Time	0-20s
Range of Throughput	0-1000kbps
Number of Users ASs	137
Number of Users Countries	31
Number of Web Services ASs	1021
Number of Web Services Countries	74



(a) Service users' location distribution



(b) Web services' location distribution

Figure 6: Location distribution: (a) service users' location distribution, with 339 service users distributed in 31 countries; (b) Web services' location distribution, with 5825 Web services distributed in 74 countries. There are 1,974,675 invocation records between the users in (a) and the services in (b).

approach. These two metrics measure the proximity between the predicted values and the real values [36]. MAE is denoted as

$$MAE = \frac{\sum_{ij} \left| \hat{r}_{ij} - r_{ij} \right|}{N}, \tag{8}$$

and the mathematical expression of NMAE is

$$NMAE = \frac{MAE}{\sum_{ij} r_{ij}/N}. \tag{9}$$

Here, N is the number of predicted QoS values, $\hat{r}_{ij}$ is the predicted QoS values, and r_{ij} is the real QoS values in the dataset. Smaller values of MAE and NMAE will lead to a better performance of the model.

5.3. Performance Comparison. To evaluate the performance of our LANFM model, the following baseline approaches are introduced for comparison. The baseline methods vary from memory-based collaborative filtering approaches and model-based collaborative filtering approaches to hybrid collaborative filtering approaches:

(1) UIPCC [4]: this method unifies the user-based collaborative filtering approach and item-based collaborative filtering approach to predict QoS for users. It is based on finding similar users and similar services

(2) PMF (probabilistic matrix factorization) [37]: this method assumes the data distribution is a Gaussian distribution. It factorizes the user-service invocation matrix to user-latent matrix and service-latent matrix. Then it uses the multiplication of these two latent matrices to predict QoS for users

(3) NMF (nonnegative matrix factorization) [38]: this method also exploits the user-service invocation matrix to obtain user-latent matrix and service-latent matrix. But it adds a constraint that the factorized latent matrices should be nonnegative

(4) NIMF (neighborhood-integrated matrix factorization) [5]: this approach firstly computes the similarities between any two users. Then, it combines the similar users' information and QoS records to conduct matrix factorization. Finally, it predicts QoS for users

(5) EFMPred (embedding based factorization machine) [14]: this approach firstly embeds the user id and service id to vectors. Then, it uses factorization machine to predict QoS for users

(6) RegionKNN (region K nearest neighbors) [27]: this approach firstly clusters users and services on the basis of the location information and QoS values. Then it employs the clustering results to predict QoS for users

(7) LACF (location-aware collaborative filtering) [12]: this approach incorporates users' and services' locations and employs a location-aware collaborative filtering approach to predict QoS for users

(8) LBR (location-based regularization) [28]: this approach uses the local connectivity between users and combines regularization terms in classical matrix factorization framework to predict QoS for users

(9) HMF (hierarchical matrix factorization) [29]: this approach firstly groups users and services according to the location information. Then, it unifies the results from local matrix factorization and global matrix factorization to predict QoS for users

In reality, only a portion of services are invoked by users. Thus, the invocation matrix between users and services is considerably sparse. We divide the dataset into training set and testing set by randomly removing a proportion of elements from the invocation matrix between users and services. For instance, if we remove 90% elements from the

Table 7: Quality of services prediction accuracy comparison.

QoS Properties	Methods	Matrix Density=10%		Matrix Density=20%		Matrix Density=80%		Matrix Density=90%	
		MAE	NMAE	MAE	NMAE	MAE	NMAE	MAE	NMAE
Response Time (0-20s)	UIPCC	0.5842	0.6433	0.4514	0.4970	0.3475	0.3822	0.3443	0.3785
	LACF	0.5612	0.6181	0.4778	0.5262	0.3692	0.4062	0.3637	0.3995
	RegionKNN	0.5491	0.6048	0.5155	0.5677	0.4950	0.5446	0.4860	0.5338
	PMF	0.4865	0.5361	0.4305	0.4743	0.3751	0.4123	0.3733	0.4098
	NMF	0.4774	0.5261	0.4269	0.4703	0.3723	0.4093	0.3705	0.4068
	NIMF	0.4792	0.5281	0.4202	0.4630	0.3665	0.4029	0.3677	0.4037
	LBR	0.4806	0.5293	0.4301	0.4737	0.3761	0.4138	0.3736	0.4103
	HMF	0.4815	0.5302	0.4298	0.4732	0.3734	0.4110	0.3698	0.4070
	EFMPred	0.3878	0.4083	0.3332	0.3606	0.2641	0.2910	0.2599	0.2825
	LANFM	*0.3607*	*0.3966*	*0.3247*	*0.3571*	*0.2635*	*0.2903*	*0.2577*	*0.2823*
	Improve	*6.98%*	*2.85%*	*2.55%*	*0.97%*	*0.24%*	*0.25%*	*0.84%*	*0.06%*
Throughput (0-1000kbps)	UIPCC	22.3274	0.4700	18.8646	0.3966	13.5460	0.2845	13.0623	0.2776
	LACF	19.4303	0.4087	16.4495	0.3459	12.4200	0.2609	12.1073	0.2545
	RegionKNN	24.8487	0.5226	24.0169	0.5050	24.0414	0.5050	23.9013	0.5023
	PMF	15.9794	0.3362	13.9052	0.2924	12.1408	0.2551	11.9442	0.2520
	NMF	15.5678	0.3275	13.5386	0.2847	11.9260	0.2506	11.7964	0.2489
	NIMF	15.1393	0.3185	13.1799	0.2772	11.8641	0.2493	11.7977	0.2489
	LBR	15.4431	0.3248	13.6455	0.2869	12.1012	0.2542	11.9560	0.2513
	HMF	15.7076	0.3304	13.5961	0.2859	11.6637	0.2450	11.5562	0.2429
	EFMPred	13.2966	0.2811	11.4295	0.2417	8.6713	0.1829	8.3839	0.1773
	LANFM	*12.8365*	*0.2714*	*11.0560*	*0.2338*	*8.5160*	*0.1796*	*8.2561*	*0.1746*
	Improve	*3.46%*	*3.45%*	*3.27%*	*3.27%*	*1.79%*	*1.80%*	*1.52%*	*1.52%*

invocation matrix, then these 90% elements are regarded as testing set; the rest of 10% elements are treated as training set.

The performance comparison results between our LANFM model and other baseline approaches are shown in Table 7. It could be found that LANFM always achieves the best performance in both evaluation metrics (i.e., MAE and NMAE), no matter whether the QoS property is response time or throughput. Specifically, the performance of matrix factorization-based methods (e.g., PMF, NMF, NIMF, LBR, and HMF) is better than memory-based methods (e.g., UIPCC, LACF, and RegionKNN) due to learning the latent factors. In addition, factorization machine-based methods (e.g., EFMPred and LANFM) outperform the matrix factorization-based methods due to learning the cross features of the user information and the service information. Furthermore, LANFM outperforms EFMPred as a result of considering more important features (e.g., users' location information and services' location information). Compared with EFMPred, LANFM has a performance improvement of 0.06% to 6.98% in response time and a performance improvement of 1.52% to 3.46% in throughput. Moreover, when the training data is sparser, the performance improvement is more obvious. In reality, the data we obtain is very sparse. The observation demonstrates that our LANFM model can be applied to sparse data. Here, only a part of the results (e.g., 10%, 20%, 80%, and 90%) is presented in the table. The experimental results under all matrix density will be introduced in Section 5.4.

5.4. Impact of Matrix Density. Matrix density is a significant parameter that affects the accuracy of QoS prediction. It means how many historical invocation records between users and services we can exploit to help predict QoS values. The density of the invocation matrix is changed from 10% to 90% to study the effect of matrix density. The step size is set as 10%. In this experiment, the dimension of embedding vector for response time is 30. With respect to throughput, the dimension value is 1400. The batch size is 4096.

Figures 7(a) and 7(b) display the prediction performance of all methods on response time in terms of MAE and NMAE. Figures 7(a) and 7(b) show that when the matrix density is increased from 10% to 50%, the downward trend in the values of MAE and NMAE is significant. However, when we increase the density from 50% to 90%, the decrease in the values of MAE and NMAE is relatively slow. The trends of Figures 7(c) and 7(d) are the same with Figures 7(a) and 7(b). This phenomenon indicates that it is reasonable to evaluate the performance of the model by regrading matrix density as a parameter. All observations on Figure 7 demonstrate that increasing QoS information appropriately is helpful to tackle the problem of data sparsity. Further, increasing QoS information appropriately is also conducive to improve QoS prediction results.

5.5. Impact of Dimension. The dimension of embedding vectors is another parameter affecting the performance of the LANFM model. It decides how many factors are used to

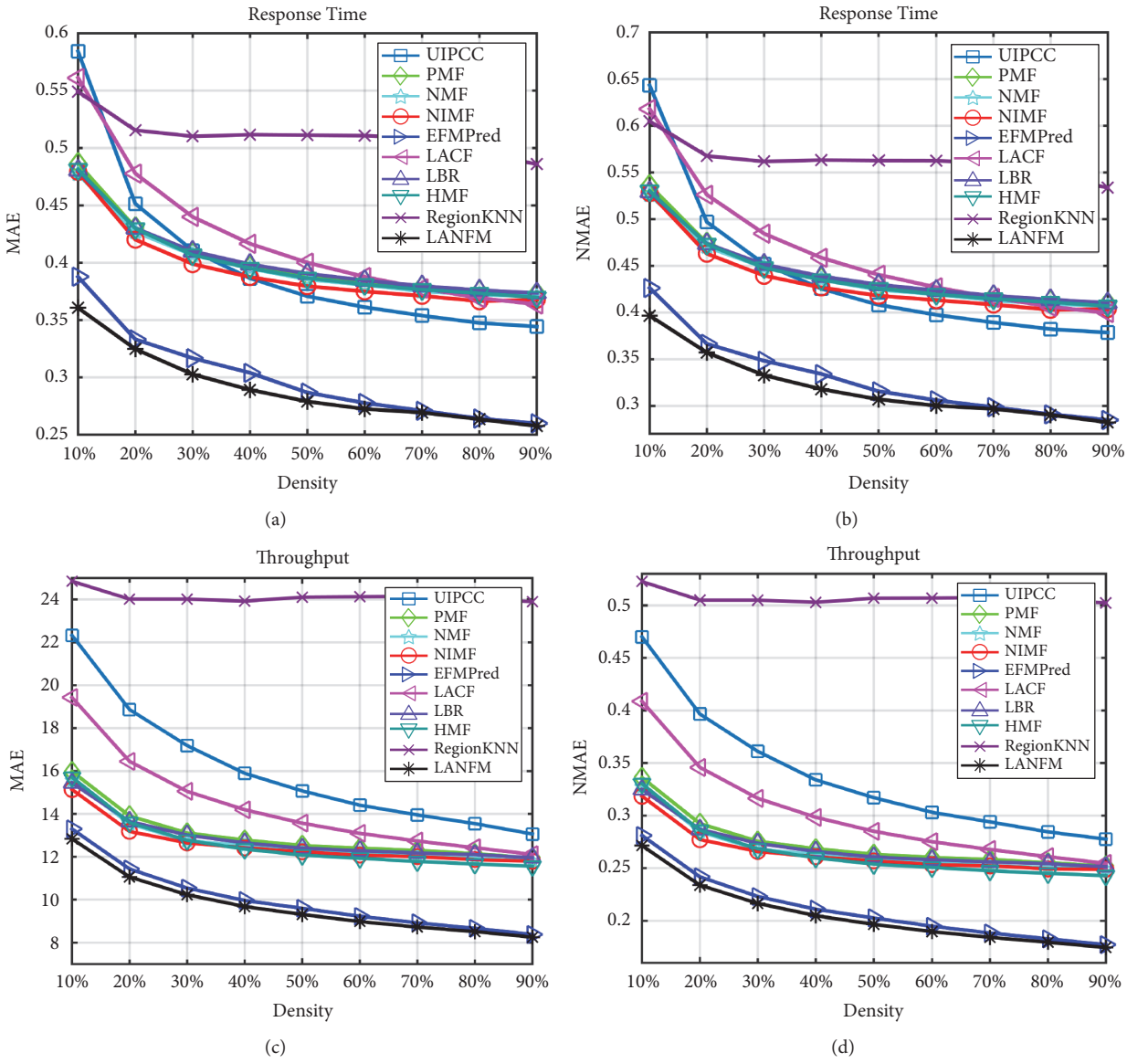


Figure 7: Impact of matrix density.

describe the feature. In this experiment, the dimension of the embedding vector for response time is changed from 5 to 50 to study the influence of the dimension. In addition, the step size is set as 5. For throughput, the dimension value is varied from 200 to 1800 and the step size is 200. The batch size is 4096.

Figures 8(a)–8(f) present the prediction performance of the LANFM model on response time, where the matrix densities are accordingly set as 10%, 50%, and 90%. The experimental results analyses of response time are as follows: (1) for 10%, the MAE value and NMAE value increase with the increment of dimension. This observation shows that when the matrix is considerably sparse, a small dimension may be suitable to enhance the QoS prediction accuracy. (2) For 50% and 90%, the MAE value and NMAE value exhibit a downward tendency at the beginning and then present an upward tendency. The lowest MAE and NMAE values are achieved at dimension=30 for 50% and dimension=35 for 90%, respectively. This observation shows that when the matrix is dense, a relatively large dimension can better improve the accuracy of QoS prediction. Nevertheless, if the dimension is set too large, it may induce the problem of overfitting, leading to poor prediction results.

Figures 8(g)–8(l) show the prediction performance of the LANFM model on throughput, where the matrix densities are correspondingly set as 10%, 50%, and 90%. Note that the scale of dimension is divided by 100 for better display. The experimental results analyses of throughput are as follows: (1) for 10% and 90%, the MAE and NMAE values of throughput decrease at first and then increase. The best MAE and NMAE values can be achieved at 600 and 1400, respectively. (2)

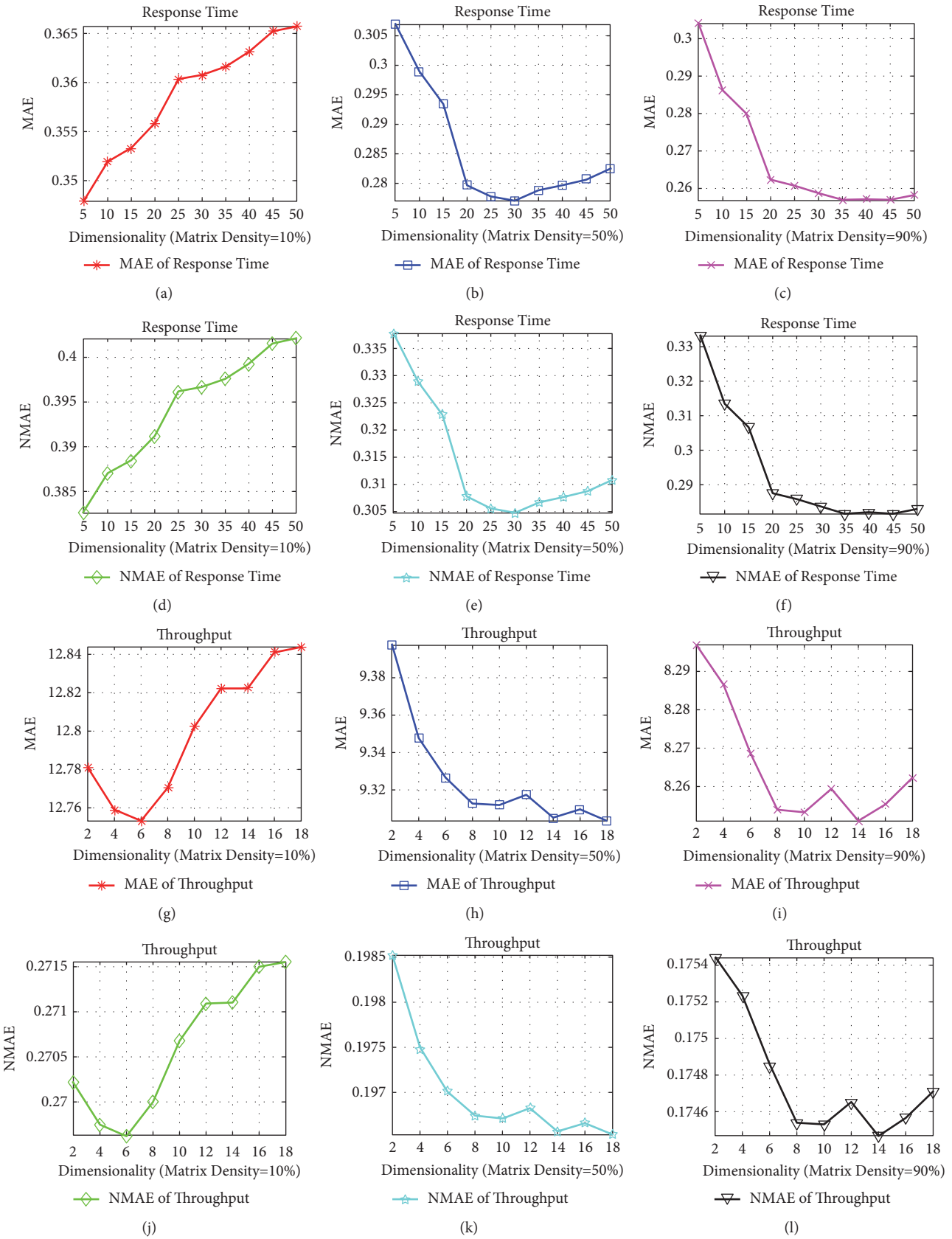


Figure 8: Impact of embedding vector dimension.

For 50%, it indicates that the MAE and NMAE values of throughput show an overall downward trend, when the dimension of the embedding vector grows from 200 to 1800. Specifically, when the dimension is from 200 to 1400, the MAE and NMAE values decline quickly, and then it gradually converges. The MAE and NMAE values oscillate around the local minimum, when varying the dimension from 1400 to 1800. The first minimum can be obtained when the dimension is 1400. Although we can also obtain the minimum at 1800, the lager dimension requires greater time complexity.

The observations on Figures 8(g)–8(l) indicate that when the matrix is very sparse, a small dimension of embedding vector is better to generate accurate QoS values. While the matrix is dense, a relatively large dimension can improve the accuracy of QoS prediction. Notice that the dimension of response time is smaller than that of throughput, since the scale of throughput value is larger than that of response time.

5.6. Impact of Batch Size. Batch size defines the number of training samples to be shown to the network before a weight update can be performed. The batch size is determined by comprehensively considering the training time and the convergence speed. When the batch size is set larger, the training speed of each epoch is faster. But the convergence speed is slower. To study the effect of the batch size, we change the batch size from 64 to 5120 for both of the QoS properties. In this experiment, the densities of the invocation matrix are set as 10%, 50%, and 90%. The dimension of embedding vector for response time is 30. For throughput, it is set as 1400. It is worthwhile to note that 5120 is the maximum memory of our GPU.

The prediction performance of the LANFM model on response time is presented in Figures 9(a)–9(f), and we can observe the following: (1) for 10%, the MAE value and NMAE value decrease significantly when increasing the batch size from 64 to 1024. While it is from 1024 to 5120, both values oscillate around the local minimum. We can obtain the best value at 4096. (2) For 50%, the MAE value and NMAE value decrease significantly when increasing the batch size from 64 to 1024. While the batch size is from 1024 to 3072, the downward speeds of MAE value and NMAE value become slow. Finally, both of MAE and NMAE values converge. This observation demonstrates that when the batch size is small, enlarging the batch size can enhance the QoS prediction accuracy greatly. When it comes to a relatively large batch size, the improvement is less obvious. (3) For 90%, the MAE and NMAE values decline quickly when the batch size is from 64 to 2048 and then up to the minimum at batch size = 4096. The observations indicate that when the matrix is quite sparse, a small batch size (e.g., 1024) is quite useful for improving the accuracy of QoS prediction. While the matrix is dense, a lager batch size (e.g., 4096) is more useful to upgrade the accuracy of QoS prediction.

As can be seen in Figures 9(g)–9(l), when increasing the batch size from 64 to 3072, the MAE and NMAE values of throughput decline quickly. When the batch size is from 3072 to 5120, the MAE and NMAE values gradually converge. The observation indicates that when the batch size is small, enlarging it can greatly improve the QoS prediction accuracy. However, after exceeding a certain threshold (e.g., batch size=3072), the enhancement is not so obvious.

6. Conclusion

This paper proposes a location-aware factorization machine approach by leveraging the embedding technique in neural networks. Firstly, the location information of services and users is taken into account. Secondly, the user information and the service information are expressed as embedding vectors to mine the potential relationships between users and services. Finally, the inner products of embedding vectors, along with the weighted sum of feature vectors, are utilized to perform QoS prediction. There are three advantages to employ neural network techniques and factorization machine model: (1) the dimension of original input feature vectors can be reduced; (2) the problem of large data sparsity can be solved; and (3) the time complexity of our LANFM model is linear. That is to say, our LANFM model is able to solve the three shortcomings: high dimension, high time complexity, and high implementation expense. Therefore, the scalability of our LANFM model is good. It is applicable to large scale datasets. A series of comprehensive experiments are carried out on the WSDream dataset to verify the effectiveness of our LANFM model. First of all, we evaluate the performance of our approach and other state-of-the-art baseline approaches under different matrix densities, which demonstrates that our LANFM model always achieves the best performance. Then, we study the impact of the dimension of the embedding vector to determine how large it should be, which indicates that, for the two QoS properties, when the matrix is very sparse, a relatively small embedding size is good to enhance the performance; when the matrix is dense, a relatively large dimension can better improve the accuracy of QoS prediction. Finally, we investigate the effect of the batch size, which is a powerful parameter that affects the performance of optimization algorithm. The results show that, for the response time, when the matrix is quite sparse, a relatively small batch size is useful for improving the prediction performance; while the matrix is dense, a relatively large batch size is more conducive to improve the QoS prediction accuracy. For the throughput, when the batch size is relatively small, increasing the batch size will improve the predictive performance. While a certain threshold is exceeded, the improvement is less distinct.

In reality, response time and throughput are dynamically changing with the network environment, so the time factor should be considered. Therefore, we will attempt to build a more powerful model, which considers the time information, to predict QoS values for users in future work. Additionally, other QoS attributes (e.g., reliability, availability, and failure probability) are rarely studied in previous work. Thus, we

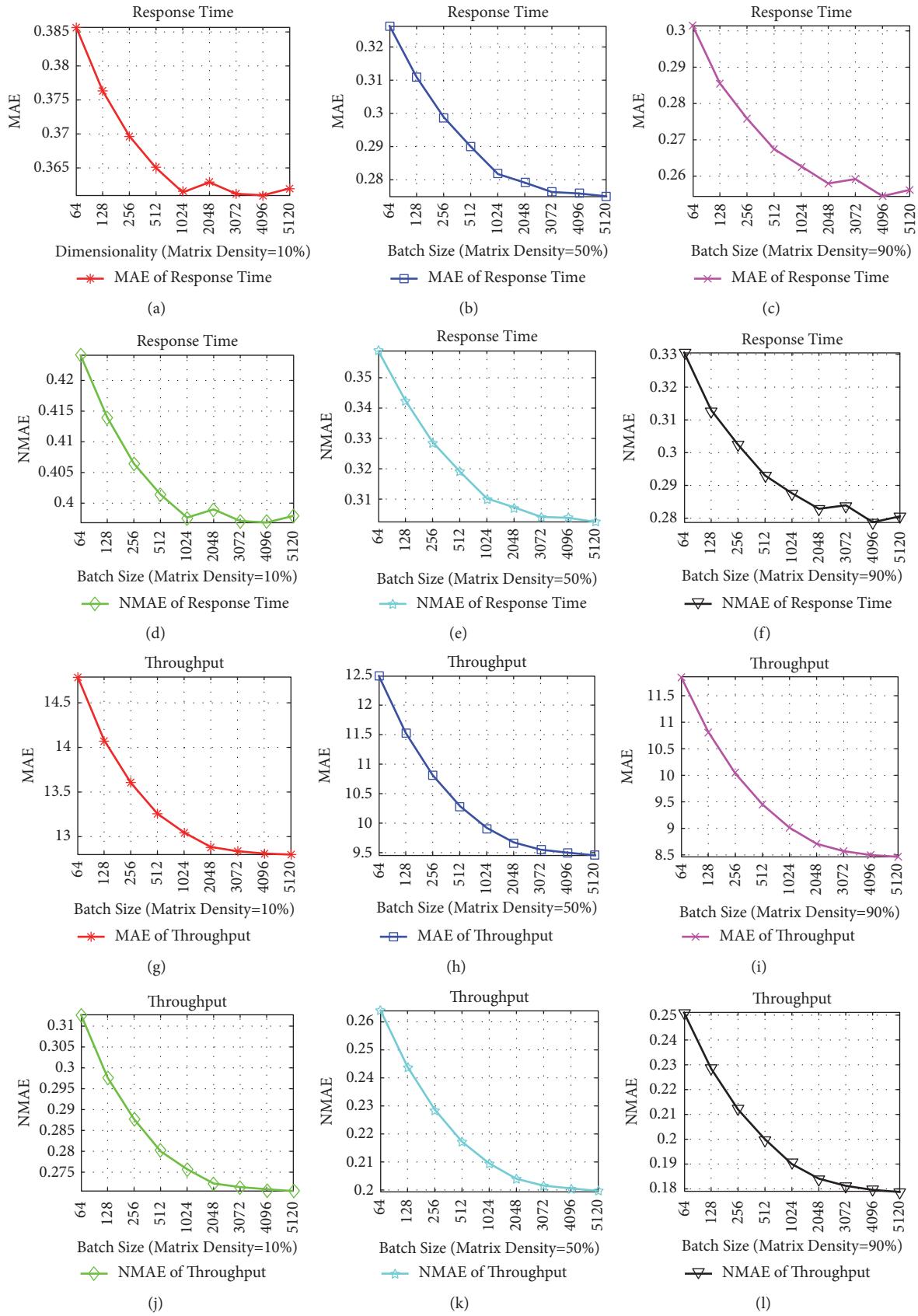


Figure 9: Impact of batch size.

would like to investigate other QoS attributes in our next work.

Data Availability

The QoS data used to support the findings of this study can be accessed publicly in the Website https://wsdream.github.io/dataset/wsdream_dataset1.html.

Disclosure

Zibin Zheng is the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The paper was supported by the National Key Research and Development Program (2017YFB0202201), the National Natural Science Foundation of China (61702568; U1711267), the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (2017ZT07X355), and the Fundamental Research Funds for the Central Universities under Grant no. 17lgpy117.

References

[1] L.-J. Zhang, H. Cai, and J. Zhang, *Services Computing*, Springer, 2007.

[2] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, "QoS-aware middleware for web services composition," *IEEE Transactions on Software Engineering*, vol. 30, no. 5, pp. 311–327, 2004.

[3] S. Chen, Y. Peng, H. Mi, C. Wang, and Z. Huang, "A cluster feature based approach for QoS prediction in web service recommendation," in *Proceedings of the IEEE Symposium on Service-Oriented System Engineering (SOSE)*, pp. 246–251, IEEE, Bamberg, Germany, 2018.

[4] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "QoS-aware web service recommendation by collaborative filtering," *IEEE Transactions on Services Computing*, vol. 4, no. 2, pp. 140–152, 2011.

[5] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Collaborative web service Qos prediction via neighborhood integrated matrix factorization," *IEEE Transactions on Services Computing*, vol. 6, no. 3, pp. 289–299, 2013.

[6] K. Su, L. L. Ma, B. Xiao, and H. Q. Zhang, "Web service QoS prediction by neighbor information combined non-negative matrix factorization," *Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology*, vol. 30, no. 6, pp. 3593–3604, 2016.

[7] H. Wu, K. Yue, B. Li, B. Zhang, and C.-H. Hsu, "Collaborative QoS prediction with context-sensitive matrix factorization," *Future Generation Computer Systems*, vol. 82, pp. 669–678, 2018.

[8] L. Kuang, L. Yu, L. Huang et al., "A personalized QoS prediction approach for CPS service recommendation based on reputation and location-aware collaborative filtering," *Sensors*, vol. 18, no. 5, p. 1556, 2018.

[9] J. Liu, M. Tang, Z. Zheng, X. Liu, and S. Lyu, "Location-aware and personalized collaborative filtering for web service recommendation," *IEEE Transactions on Services Computing*, vol. 9, no. 5, pp. 686–699, 2016.

[10] K. Lee, J. Park, and J. Baik, "Location-based web service QoS prediction via preference propagation for improving cold start problem," in *Proceedings of the IEEE International Conference on Web Services (ICWS)*, pp. 177–184, IEEE, New York, NY, USA, 2015.

[11] X. Chen, Z. Zheng, Q. Yu, and M. R. Lyu, "Web service recommendation via exploiting location and QoS information," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1913–1924, 2014.

[12] M. Tang, Y. Jiang, J. Liu, and X. Liu, "Location-aware collaborative filtering for qos-based service recommendation," in *Proceedings of the 19th International Conference on Web Services (ICWS)*, pp. 202–209, IEEE, 2012.

[13] S. Rendle, "Factorization machines," in *Proceedings of the 10th International Conference on Data Mining (ICDM)*, pp. 995–1000, IEEE, Australia, 2010.

[14] Y. Wu, F. Xie, L. Chen, C. Chen, and Zheng Z., "An embedding based factorization machine approach for web service qos prediction," in *Proceedings of the International Conference on Service-Oriented Computing*, pp. 272–286, Springer, 2017.

[15] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pp. 43–52, Morgan Kaufmann Publishers Inc., 1998.

[16] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 230–237, ACM, 1999.

[17] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, 2003.

[18] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th International Conference on World Wide Web (WWW)*, pp. 285–295, ACM, 2001.

[19] J. Wang, A. P. D. Vries, and M. J. Reinders, "Unifying user-based and item-based collaborative filtering approaches by similarity fusion," in *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, (SIGIR)*, pp. 501–508, Seattle, Wash, USA, 2006.

[20] H. Sun, Z. Zheng, J. Chen, and M. R. Lyu, "Personalized web service recommendation via normal recovery collaborative filtering," *IEEE Transactions on Services Computing*, vol. 6, no. 4, pp. 573–579, 2013.

[21] W. Xiong, B. Li, L. He, M. Chen, and J. Chen, "Collaborative web service QoS prediction on unbalanced data distribution," in *Proceedings of the 2014 International Conference on Web Services (ICWS)*, pp. 377–384, IEEE, Anchorage, AK, USA, 2014.

[22] Y. Ma, S. Wang, P. C. Hung, C. H. Hsu, Q. Sun, and F. Yang, "A highly accurate prediction algorithm for unknown web service QoS value," *IEEE Transactions on Services Computing*, vol. 9, no. 4, pp. 511–523, 2016.

[23] J. Xu, Z. Zheng, and M. R. Lyu, "Web service personalized quality of service prediction via reputation-based matrix factorization," *IEEE Transactions on Reliability*, vol. 65, no. 1, pp. 28–37, 2016.

[24] X. Luo, M. Zhou, Y. Xia, and Q. Zhu, "Predicting web service qos via matrix-factorization-based collaborative filtering under non-negativity constraint," in *Proceedings of the 23rd Wireless and Optical Communication Conference (WOCC)*, pp. 1–6, IEEE, Newark, NJ, USA, 2014.

[25] Z. Chen, L. Shen, D. You, and F. Li, "A user dependent Web service QoS collaborative prediction approach using neighborhood regularized matrix factorization," in *Proceedings of the 20th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 316–321, IEEE, China, 2016.

[26] W. Lo, J. Yin, S. Deng, Y. Li, and Z. Wu, "An extended matrix factorization approach for QoS prediction in service selection," in *Proceedings of the International Conference on Services Computing (SCC)*, pp. 162–169, IEEE, Honolulu, HI, USA, 2012.

[27] X. Chen, X. Liu, Z. Huang, and H. Sun, "Regionknn: a scalable hybrid collaborative filtering algorithm for personalized web service recommendation," in *Proceedings of the 8th International Conference on Web Services (ICWS)*, pp. 9–16, IEEE, 2010.

[28] W. Lo, J. Yin, S. Deng, Y. Li, and Z. Wu, "Collaborative web service QoS prediction with location-based regularization," in *Proceedings of the 19th International Conference on Web Services (ICWS)*, pp. 464–471, IEEE, Honolulu, Hawaii, USA, 2012.

[29] P. He, J. Zhu, Z. Zheng, J. Xu, and M. R. Lyu, "Location-based hierarchical matrix factorization for Web service recommendation," in *Proceedings of the 21st International Conference on Web Services, (ICWS)*, pp. 297–304, IEEE, 2014.

[30] J. Li, T. Sellis, J. S. Culpepper, Z. He, C. Liu, and J. Wang, "Geo-social influence spanning maximization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 8, pp. 1653–1666, 2017.

[31] G. Zhang, M. Tang, S. Cheng et al., "P2P traffic optimization," *Science China Information Sciences*, vol. 55, no. 7, pp. 1475–1492, 2012.

[32] S. Golson, "One-hot state machine design for fpgas," in *Proceedings of the 3rd Annual PLD Design Conference & Exhibit*, vol. 1, 1993.

[33] X. He and T. Chua, "Neural factorization machines for sparse predictive analytics," in *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 355–364, Shinjuku, Tokyo, Japan, 2017.

[34] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of the COMPSTAT'2010*, pp. 177–186, Springer, Berlin, Germnay, 2010.

[35] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pp. 448–456, 2015.

[36] C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance," *Climate Research*, vol. 30, no. 1, pp. 79–82, 2005.

[37] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," *NIPS*, vol. 1, no. 1, pp. 1257–1264, 2007.

[38] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

Hindawi
Complexity
Volume 2019, Article ID 2950287, 23 pages
https://doi.org/10.1155/2019/2950287

Research Article

An Effective Algorithm for Video-Based Parking and Drop Event Detection

Gang Li, Huansheng Song, and Zheng Liao

School of Information Engineering, Chang'an University, Xi'an 710064, Shaanxi, China

Correspondence should be addressed to Gang Li; 1723915372@qq.com

Received 13 November 2018; Revised 11 February 2019; Accepted 18 March 2019; Published 7 April 2019

Guest Editor: Ke Deng

Real-time and accurate detection of parking and dropping events on the road is important for the avoidance of traffic accidents. The existing algorithms for detection require accurate modeling of the background, and most of them use the characteristics of two-dimensional images such as area to distinguish the type of the target. However, these algorithms significantly depend on the background and are lack of accuracy on the type of distinction. Therefore, this paper proposes an algorithm for detecting parking and dropping objects that uses real three-dimensional information to distinguish the type of target. Firstly, an abnormal region is initially defined based on status change, when there is an object that did not exist before in the traffic scene. Secondly, the preliminary determination of the abnormal area is bidirectionally tracked to determine the area of parking and dropping objects, and the eight-neighbor seed filling algorithm is used to segment the parking and the dropping object area. Finally, a three-view recognition method based on inverse projection is proposed to distinguish the parking and dropping objects. The method is based on the matching of the three-dimensional structure of the vehicle body. In addition, the three-dimensional wireframe of the vehicle extracted by the back-projection can be used to match the structural model of the vehicle, and the vehicle model can be further identified. The 3D wireframe of the established vehicle is efficient and can meet the needs of real-time applications. And, based on experimental data collected in tunnels, highways, urban expressways, and rural roads, the proposed algorithm is verified. The results show that the algorithm can effectively detect the parking and dropping objects within different environment, with low miss and false detection rate.

1. Introduction

With the increasing demands of traffic transportation in modern life, such as express delivery and logistics, the number of motor vehicles in the city continues to rise. The increase in the number of motor vehicles has caused numerous problems, such as parking and dropping incidents that reduce road traffic efficiency [1]. Therefore, accurately detecting the parking and dropping events on the road in real time is a key factor to ensure a safety-of-life traffic system [2].

Parking and throwing objects are static targets in traffic scenes. The detection algorithms for such targets in intelligent-traffic-incident-detection systems developed at home and abroad are mainly divided into two steps: target area detection and target type differentiation.

There are two methods for target area detection: tracking method and nontracking method.

The tracking method detects the stationary target by analyzing the characteristics of the foreground target trajectory. For example, Bevilacqua et al. [3] firstly obtain the foreground target by background difference; secondly, use the optical flow method to track the target; finally, analyze the displacement of the target center position. If the displacement of the target center position has been moving within a small range for a certain period of time, it is considered that a parking event has occurred. This method is simple to implement but there is a problem if the parking is detected during the parking period. Bing-Fei Wu et al. [4] proposed a tunnel event detection system. First, the background extraction is performed; then the foreground target is obtained by subtracting the background; finally, the target is tracked. Considering that the movement of the parked vehicle is small, the average distance of the vehicle is calculated using the trajectory line to determine the parking

event. Guler et al. [5] used the object tracker and the scene description layer (background) to detect the stopped vehicle. The basic idea of the algorithm is to use the object tracker to track the target in the scene. When a pixel is detected as a still pixel, it is compared with the scene description layer. When the two are similar, the pixel is a static target. The likelihood will decrease and vice versa. Although the method uses the samples in the i-LIDS for testing, the accuracy can be met, including the parking lot scene and the legacy package scene. But there are still two shortcomings: on the one hand, the rapidly changing background is considered a parking event. On the other hand, when the scene description layer itself contains a static vehicle, the area where the vehicle leaves is mistakenly detected as the area where the parking event occurs. In a traffic scene with a large traffic flow, it is difficult to extract a background image that does not contain any vehicle. Zhang Beibei et al. [6] used the particle filter algorithm combined with the OSTU threshold to detect the stationary target. The biggest problem of this method is the selection of the threshold and the false positive caused by the void after segmentation. Akhawaji R et al. [7] used a mixed Gaussian model to model the background and use Kalman filtering to detect stationary targets in the forbidden area. The algorithm is sensitive to illumination and is easy to lose targets when traffic is heavy. He T et al. [8] used special GPS points for map matching and track indexing, then simulates normal trajectories, extracts features, and uses distributed testing to detect illegal parking.

The nontracking method mainly relies on background modeling and analysis of foreground pixel time series features to detect stationary target regions. For example, Fatih Porikli et al. [9] used the double background method to detect the parking litter. This method does not use any tracking technology to detect abnormal events only by subtracting the background. The basic idea is as follows: Firstly, two mixed time Gaussian models are used to establish two backgrounds with different time constants (short background and long background), and the online Bayesian update mechanism is used to update the two backgrounds in real time. Among them, the short background describes the most recent target from motion to still, and the long background describes the real background of the scene. When there are foreground pixels in the scene, compare them with the two backgrounds, respectively. When the pixel is very similar to the short background and has a large difference from the long background, the pixel is considered to be a static target pixel. If a pixel is continuous it has been a static target pixel for a period of time, and the pixel is marked as an abnormal pixel. This method has a certain real-time performance, but the anti-interference is poor, and the time constant when establishing a long background and a short background is difficult to determine. Stauffer et al. proposed a parking detection algorithm that uses the difference method to achieve target extraction. The algorithm has strong real-time performance. However, its biggest shortcoming is that when the target type is identified, the interference caused by other factors (such as pedestrians, bicycles, etc.) is not filtered out, resulting in an increase in the false positive rate. Zhao Min et al. [10] showed that, in order to avoid the false positive rate which is too high, firstly, the hybrid Gaussian model is used to obtain the suspected moving foreground target in the background extraction and update; then, the steady state change of the foreground target is analyzed to detect the stationary target. There are two major shortcomings of this method. First, the amount of calculation is large. Second, when the training time is too short and the background model is not fully trained, the foreground target is estimated as the background, which affects the detection of static targets.

Static targets in traffic scenes are mainly divided into parking and throwing objects, so the distinction between static targets is the distinction between parking and throwing objects. The current algorithm mainly uses the two-dimensional features of the target to identify it. For example, Wang Dianhai and Hu Hongyu [11] identify the difference between the vehicle target and the target area of the target in the foreground. However, due to the angle of the camera, etc., the target of the throwing object detected in the close scene is not much different from the target area of the vehicle detected in the distant view. Based on the detection of the target area, Mu Chunyang [12] used Hough ellipse fitting, wheel circularity and compact feature extraction to identify the parking events.

Today's machine learning [13] applications on images do have very good performance, but the traditional methods we use still have advantages: (1) Today, when machine learning is prevalent, it does not mean abandoning traditional methods. In-depth study of traditional methods is something we have been doing and is valuable. (2) In the future, we hope to continue to develop in the direction of embeddedness. The expensive cost of machine learning limits the popularity of the methods we propose in the paper. Therefore, we have chosen a method based on three-dimensional information to classify vehicles.

Through the survey of research on video-based parking and dropping objects, the key issues found in most algorithms for detecting parked and discarded objects are mainly in two aspects. The first issue is how to detect the target, which is the core part of the algorithm. Tracking and nontracking methods are generally used to detect the target area. Both methods need to extract and update the background. However, under complicated traffic scenes such as low visibility, large traffic flow, and intense lighting changes, it is difficult to extract an ideal background image. The second one is how to distinguish the target type. When distinguishing the target type, the two-dimensional feature of the target is often used, but the imaging process of the camera is a process of dimensional reduction. In this process, the target will undergo significant scale changes and geometric deformation. Therefore, these methods of using image features to identify targets have significant limitations. In view of these two shortcomings, it is of great theoretical and practical value to study algorithms for detecting parking and dropping objects that do not rely on background and use real three-dimensional information for target recognition.

In this paper, the above problems are studied and a new method is proposed. The real-time video collected by the camera is used as the data source. The image analysis and processing program is used to realize the automatic detection

and feedback of parking and litter events. It is mainly divided into the following three steps. Firstly, based on the state evolution, the initial determination of the abnormal region in the image is carried out. Secondly, the two-way tracking and eight-neighbor seed filling algorithm is used to segment the parking and the drop area in the image. Finally, the three-dimensional information is used to distinguish the target.

2. Preliminary Determination of Abnormal Regions in Images

The detection of abnormal regions is the core of the whole algorithm, so choosing an appropriate detective algorithm is the first problem to consider. Current algorithms for detection are too dependent on the background and computationally intensive. We will use status change to detect abnormal regions, which can effectively avoid the above two shortcomings.

The abnormal area refers to the image area where the steady state changes. The basic idea of the algorithm is as follows: Firstly, the image is preprocessed to highlight useful information, which lays a foundation for improving the accuracy of subsequent detection. Secondly, the detection of abnormal regions is carried out based on the status change, and solutions are given for some of the shortcomings. Finally, an improved algorithm for detection is proposed, and the results before and after the improvement are compared and analyzed.

2.1. Image Preprocessing. When the video image is captured, the camera will be affected by factors such as illumination changes and noise pollution, which will affect the captured image. In order not to affect the result of the algorithm, the image will be preprocessed by image enhancement, edge extraction, and median filtering.

2.1.1. Image Enhancement. In the case of night or smog, the contrast and color of the collected traffic video images will be degraded, and a lot of useful information will be covered, which is very unfavorable for the subsequent algorithm. Among the algorithms for image enhancement, the gray-level section transforms based on gray level transformation which has been widely used due to its advantages such as simplicity and diverse transformation functions [14–16]. In this paper, the original image is processed by a three-stage linear transformation, as shown in Figure 1. The expression as follows:

$$g(x,y) = \begin{cases} c & 0 \le f(x,y) \le a \\ \dfrac{d-c}{b-a}\left[f(x,y)-a\right]+c & a \le f(x,y) \le b \\ d & b \le f(x,y) \le M \end{cases} \quad (1)$$

In Figure 1, f(x, y) represents the original gray value at (x, y), g(x, y) represents the gray value at (x, y) after image enhancement, and M is the maximum gray value of 255. And Figure 2 shows the original image and the enhanced image.

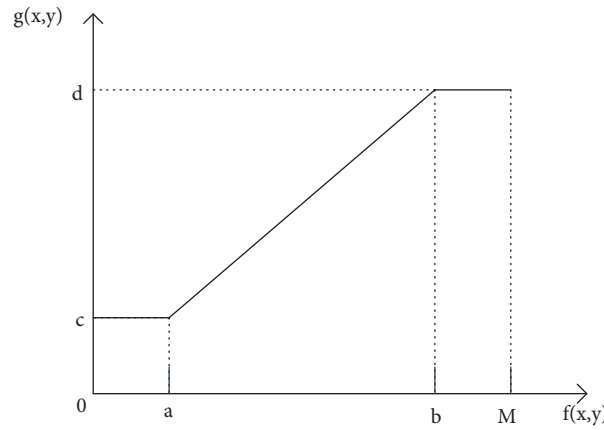


Figure 1: Linear transformation of gray scale.

Comparing the two images, it can be seen that the contrast of the enhanced image is significantly improved.

2.1.2. Edge Extraction. The edge is the most basic feature of the image, which is invariant to changes in light, so in order to reduce the effect of light on the detection result, edge extraction is applied after image enhancement. There are many classic operators for edge detection [17], after considering the detection effect, real-time performance, and arithmetic speed. We employ a simplified first-order differential operator that uses local differences to find the edges of the image. Its expression is as follows:

$$E(x,y) = \max\left(\left(\left|f(x,y) - f(x,y+1)\right|\right), \left(\left|f(x,y) - f(x+1,y)\right|\right)\right) \quad (2)$$

f(x, y) represents the original gray picture and E(x, y) represents the edge enhancement gray picture. The experimental results of edge enhancement of the video image of Xi'an South Second Ring by the operator are shown in Figure 4(c).

2.1.3. Median Filtering. In the process of acquisition, transmission, and storage, any process may introduce noise, and the presence of noise seriously affects the result of edge enhancement. Therefore, it is necessary to denoise the detected edge image. The median filter [18] also filters out noise while maintaining the edges of the image. The basic principle is as follows: select a symmetrical area centering on each pixel in the image, and sort all the pixel values in the area, taking the middle pixel value as the pixel value of the current point. Figure 3 is a result of the horizontal straight type window filtering of Figure 4(c), from which it can be found that the image quality is significantly improved, and the edge of the image is well preserved while filtering noise.

2.2. Abnormal Region Determination Based on Status Change. Usually the gray value of each pixel in the image does not change for a long time. Only when the foreground target passes from the pixel area, the gray value of the point will be changed, and when the foreground target passes, the change of the pixel gray value is greater than the change caused by the

(a) Original image (b) Enhanced image

Figure 2: Original image and enhanced image.

Figure 3: Median filter effect.

environmental influence. Thus, when it is detected that there is a pixel point in the image where the gray value changes greatly, which means that the status change happen, it is determined that the foreground object exists. The detected change in the gray value may be caused by the moving target passing through the detected area or the target entering the detected area and stopping. Therefore, in order to correctly detect the parking and the dropping object, it is necessary to remove the gray value caused by the moving target passing through the detected area.

When the pixel gray value suddenly changes and returns to the initial gray value in a short time, it indicates that the moving target passes through the area, but does not stop. And when the pixel gray value suddenly changes and remains stable for a while, indicating that the pixel point is occupied by the foreground target, it is highly probable that a parking or dropping event occurs. Since a single pixel contains too little information and does not take into account the influence of surrounding pixels, the block is used as the basic processing unit.

Figure 5 shows the texture's change of a block in two situations, where the moving object passes through the detected area and the moving object enters the detected area and stops.

As can be seen from Figure 5, when the moving object passes through the detected area, the gray values of the two stable states are not much different. However, when the moving object stops in the detected area, the gray values of the two stable states are greatly different. In order to get rid of the dependence on the background, preliminary determination is made on the abnormal region in the image after comparing the difference between the two stable states of the block.

2.2.1. Basic Concept

(1) Image Segmentation. The video image has a dimension of 720∗288 and is divided into blocks of dimension 8∗6, so the image is divided into 90∗48 blocks. Block's coordinates are shown in Figure 6.

(2) SAD Value (Sum of Absolute Difference). SAD represents the sum of the absolute values of the pixel gray differences at corresponding positions between blocks. In this paper, SAD is mainly used for two aspects: one is the similarity matching between the current frame and the template frame; the other is the similarity matching between the current state and the reference state. Therefore, the SAD calculation formula of any block at time t is as follows:

$$SADT_t(m,n) = \sum_{y=0}^{h-1}\sum_{x=0}^{w-1} \left|E_t(x,y) - T_t(x,y)\right| \quad (3)$$

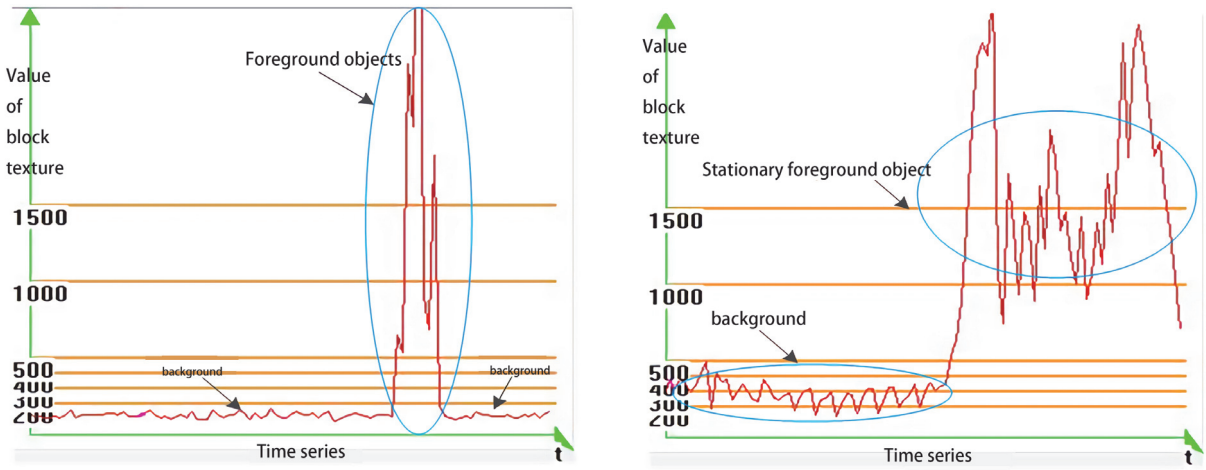
$$SADS_t(m,n) = \sum_{y=0}^{h-1}\sum_{x=0}^{w-1} \left|SR_t(x,y) - SV_t(x,y)\right| \quad (4)$$

$SADT_t(m,n)$ represents the SAD value between the current frame and the template frame at block coordinates (m,n). $SADS_t(m,n)$ represents the SAD value between the current state and the reference state at block coordinates (m,n). $E_t(x,y)$ and $T_t(x,y)$ represent the gray values of the pixel coordinates (x, y) in the current frame and the template frame. $SR_t(x,y)$ and $SV_t(x,y)$ represent gray values of pixel coordinates (x, y) in the reference state and the current state. x=0,1,. . ., w-1. y=0,1,. . ., h-1. m=0,1,. . ., C-1. n=0,1,. . ., R.

(3) Definition of Exception Block. The detected area in this article is the entire lane in the image. For any small block in the detected area, there only are two states: a road state and an abnormal state; the latter one represents parking or dropping. If a small piece maintains a state for a while, the small piece is considered to be in a steady state. The abnormal block in the

(a) Original image (b) Enhanced image (c) Edge image

Figure 4: Image preprocessing.



(a) Moving object passing through the detection area (b) Moving object enters the detection area and stops

Figure 5: Texture changes of blocks in the detection area.

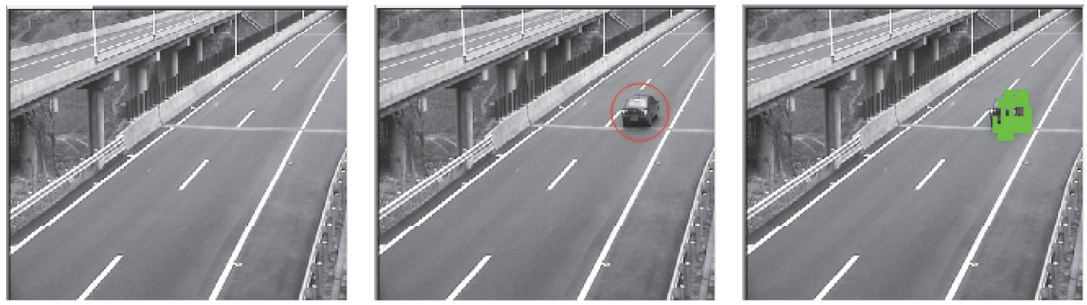
Figure 6: Block coordinate.

text refers to an image block that changes from a road stable state to an abnormal stable state.

2.2.2. Algorithm Description and Experimental Results. Based on the basic concepts, this section will introduce the core of the algorithm. According to the above method of dividing the image, a counter C(m,n) with an initial value of 0 and an abnormality flag D1(m,n) and D2(m,n) with an initial value of false are set for each block in the image, the (m,n) is the block coordinates. The abnormality flag D1(m,n) is used to mark whether the block meet the state change, and D2(m,n) is used to mark whether the block meet the forward trajectory but has no backward trajectory after bidirectional tracking of the block. First, the first frame image in the video is assigned to the template frame, and then starting from the second frame image, each image block is detected according to the following steps.

Firstly, calculate $SADT_t(m,n)$ of the block in the current frame and the corresponding block in the template frame according to formula (3). If $SADT_t(m,n)<Thsad$, then the data of the current frame of the block matches the template frame's data, and the counter C (m,n) is incremented by 1. Otherwise the counter reset and the data of the corresponding block in the template are updated with the current frame's data of the block.

Secondly, when the value of the counter C(m,n) reaches the threshold Tha, the gray value of the block in the current frame is saved. And if the threshold Tha is reached for the first time, the gray value is saved to $R_t(m,n)$; otherwise it is saved to $V_t(m,n)$.



(a) Frame 389 (b) Frame 509 (c) Frame 609

Figure 7: Test results of parking on Chongqing Expressway.

(a) Frame 12 (b) Frame 634 (c) Frame 786

Figure 8: Xi'an South Second Ring Road parking test results.

Thirdly, when the value of the counter C(m,n) reaches the threshold Thb (Thb>Tha), save the gray value of the block in the current frame to $V_t(m,n)$, and calculate $SADS_t(m,n)$ according to formula (4). If $SADS_t(m,n)<Thc$, it means that the status has not changed. Otherwise, set D1(m,n) to True, indicating that the block is an exception block. The reference state is updated with the current state regardless of whether the state changes or not; that is, $R_t(m,n)=V_t(m,n)$.

In order to verify the effectiveness of the above algorithm, the experiment was carried out in four scenarios: Chongqing Expressway, Xi'an South Second Ring Road, Xi'an South Second Ring Bus Lane, and Shanghai Fuxing Tunnel. The experimental results are as follows.

In Figures 7–11, (a) shows that no parking or dropping event, (b) the red circle in the figure indicates an abnormal event, and (c) the green image block in the figure indicates the detected abnormal region.

As can be seen from Figures 7–11, the algorithm can effectively detect abnormal blocks in the image. But through Figures 12–14, it is found that the algorithm is too sensitive to illumination, image noise, etc.

2.2.3. Existing Problems and Solutions. In order to eliminate false alarm, the algorithm are analyzed and found to have the following defects.

(1) Moving Target Texture. Some false alarms due to image noise contain less edge information. For example, in the edge-enhanced graph of the block (6, 30) in Figure 15, the gray scale of the block is enlarged to find that the texture change is smooth, and the gray scale is mainly concentrated between 0 and 13, and the total gray value of the block is low. The edge information of moving objects such as vehicles and dropping objects is more obvious. The gray-scale distribution of the block (12, 51) in Figure 16 is more dispersed, and the total gray value is significantly higher than the block (6, 30) shown in Figure 15(a).

Therefore, it can be judged whether it is an abnormal block according to the total gray value in the image block, and the calculation formula is as formula (5).

(2) Impact of Nearby Vehicles on Test Results

$$GS(m,n) = \sum_{y=0}^{h-1} \sum_{x=0}^{w-1} E(x,y) \tag{5}$$

By conducting experiments on road sections with different traffic densities, it can be found that the time required to detect anomalies on road sections with less traffic density (number of image frames) is shorter than high traffic density. As shown in Figures 7–11, it takes about 100 frames to detect anomalies in a high-speed scene in Chongqing. And it takes about 150 frames to detect anomalies in the scene of the South Second Ring Road in Xi'an.

Figure 17(a) shows the real-time video image of the South Second Ring Road in Xi'an, where the traffic density is large. Taking the red block (67, 16) as an example, Figure 17(b) shows the curve of ST and counter C during the t time. And the two curves show that there are 8 cars passing by in the period and interfering with the results of the test. This makes the counter unable to reach the threshold λ, which means that the steady state cannot be reached.

(a) Frame 2435 (b) Frame 2601 (c) Frame 2755

Figure 9: Xi'an South Second Ring Busway parking detection results.

(a) Frame 891 (b) Frame 965 (c) Frame 1067

Figure 10: Detection results of the droppings on Chongqing Expressway.

From Figure 17(b), it can be found that the counter C of the block (67, 16) reaches the higher value δ three times during the t period, as shown at three points A, B, and C. And during these three periods, the counter accumulation is due to the fact that the vehicle matches the template, so it is possible to consider connecting the counters so that the block reaches a steady state more quickly. The state in which the counter reaches δ is referred to as the potential steady state. Therefore, the difference between adjacent potential stable states in the time series can be considered, and if the difference is small, the counter is accumulated.

(3) The Effect of Slow Changes in Light on Experimental Results. The abnormal block in Figures 12(c) and 13(c) is a false alarm due to a slow change in illumination. Figure 18 is a graph showing the ST and counter C curves of a false alarm block (12, 29) in Figure 13(c).

It can be seen from Figure 18(b) that the ST of the block changes smoothly; even at the point A (red ellipse), the ST value when the counter is cleared due to exceeding the threshold Thsad is not much different from the previous ST value. As can be seen from Figure 19(b), the ST will suddenly change when the counter is cleared caused by the vehicle passing by. In this case, consider the relationship between the ST change of the E frame before the counter is cleared (in view of the calculation amount, E generally takes 200) and the current ST, and determine whether it is a false alarm caused by illumination.

In this paper, the variance of historical ST values is used to measure the change. The specific calculation formula is as follows:

$$\mu(m,n) = \frac{1}{E}\sum_{t=0}^{E} SADT_t(m,n) \tag{6}$$

$$\sigma(m,n) = \frac{1}{E}\sum_{t=0}^{E} \left|\mu(m,n) - SADT_t(m,n)\right| \tag{7}$$

μ(m, n) represents the mean of the SADT of the block (m, n) before the E frame and δ(m, n) represents the variance of the SADT. If the current SADT(m, n) of the block (m, n) exceeds the threshold Thsad, the change is slower than the SADT(m, n) of the previous E frame, and the $SADT_t$(m, n), μ(m, n), and δ(m, n) of the block (m, n) meet the formula (8). The false alarm is considered to be caused by a slow change in illumination, and the counter is directly cleared. Otherwise, it is caused by the passing of the vehicle, and the counter is accumulated.

$$\left|SADT_t(m,n) - \mu(m,n)\right| \le \alpha * \delta(m,n) + Base \tag{8}$$

$\alpha > 0$ in formula (8), the value of which changes according to the scene, *Base* is the base value, generally $\alpha = 3$, and $Base = 20$.

2.2.4. Improved Algorithm and Experimental Results. Based on the algorithm proposed in Section 2.2.2, this section comprehensively considers the edge information and the change of the history ST to improve the algorithm. StateFlag1(m, n) and StateFlag2(m, n) whose initial values are both false indicate whether the block has a potential stable state and a stable state. Use PotentialState(m, n) to save the potential stability of the block. The specific steps are as follows.

(a) Frame 2435 (b) Frame 5201 (c) Frame 5356

Figure 11: Results of detecting the dropping in the bus lane of South Second Ring Road in Xi'an.

(a) Frame 106 (b) Frame 830 (c) Frame 830

Figure 12: Shanghai Fuxing Road Tunnel Exit.

The first frame in the video is assigned to the template frame, then from the second frame, each image block is detected as follows.

Step 1. Calculate $SADT_t(m, n)$ of the block in the current frame and the corresponding block in the template frame according to formula (3), and record the $SADT_t(m, n)$.

Step 2. If $SADT_t(m, n)<Thsad$, then the current frame data of the block matches the template frame data, and the counter C(m, n) is incremented by one. Otherwise, the mean and variance of the historical SADT of the block are calculated by equations (6) and (7). If the formula (8) can be met, the counter is cleared and the data of the corresponding block in the template is updated with the current frame data of the block. If the counter reaches the threshold δ ($\delta<Tha$) and StateFlag1(m, n)=False, the gray value of the current block is saved to the potential stable state PotentialState(m, n). After the value of the counter C(m, n) is saved to PeekC(m, n), the counter is cleared, and StateFlag1(m, n) is set to True. If the counter C(m, n) reaches the threshold δ and StateFlag1(m, n)=True, the SAD between the current gray level of the block and the potential steady state PotentialState(m, n) is calculated. If SAD<Thsad, connect the counter, the counter peak value is assigned to the current counter, C(m, n)=PeekC(m, n).

Step 3. If the value of the counter C(m, n) reaches the given threshold Tha and StateFlag2(m, n)=False, it means that the counter reaches the threshold Tha for the first time, then the gray value is stored in $SR_t(m, n)$ and StateFlag2(m, n) is set to True. Otherwise it is saved in $SV_t(m, n)$.

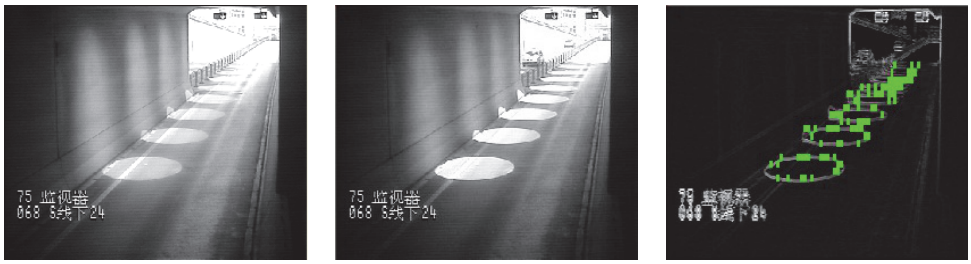
Step 4. When the value of the counter C(m, n) reaches a given threshold Thb (Thb>Tha)), $SADT_t(m, n)$ is calculated according to (4). If $SADT_t(m, n)<Thc$, this indicates that the steady state has not changed; otherwise go to Step 5. The reference state is updated with the current state regardless of whether the state changes or not, that is, $SR_t(m, n)=SV_t(m, n)$.

Step 5. Calculate the sum of the pixel values of the current block using (5). If GS(m, n)<Thd, the block's exception flag D1(m, n) is set to false; otherwise it is set to True, indicating that the block is an exception block.

In order to verify the improved algorithm, the results are compared. As shown in Figures 19, 20, and 21, the improved algorithm has significantly reduced false alarm and is more adaptable to complex environments. As can be seen from Figure 22, the original algorithm can detect the abnormality at the 786th frame, and the improved algorithm can detect the abnormality at the 755th frame. Therefore, the improved algorithm has a significant improvement in detection speed.

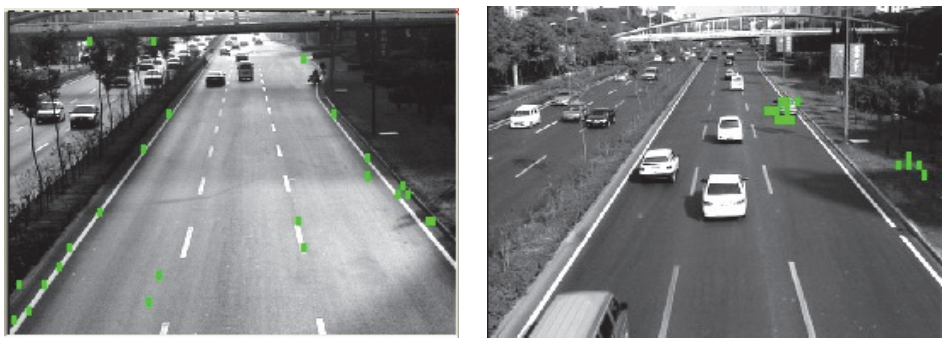
3. Segmentation of Parking and Dropping Areas in the Image

The anomalous area detected in the second section inevitably includes the effects caused by illumination changes, noise, etc. Therefore, this section first performs bidirectional tracking on the selected abnormal area, which is determined to be caused by a parking or a parachute event. Then, the eight-neighbor seed filling algorithm is used to analyze the final abnormal region to segment the parking and the dropping area.



(a) Frame 79 (b) Frame 1566 (c) Frame 1566

Figure 13: Shanghai Fuxing Road Tunnel Entrance.

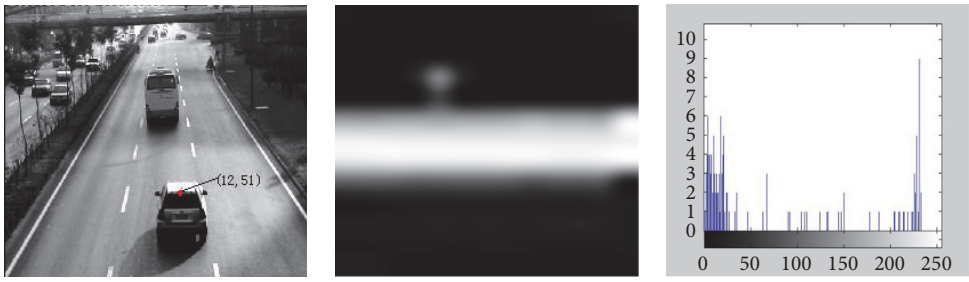


(a) Frame 1556 (b) Frame 3484

Figure 14: Xi'an South Second Ring Road.

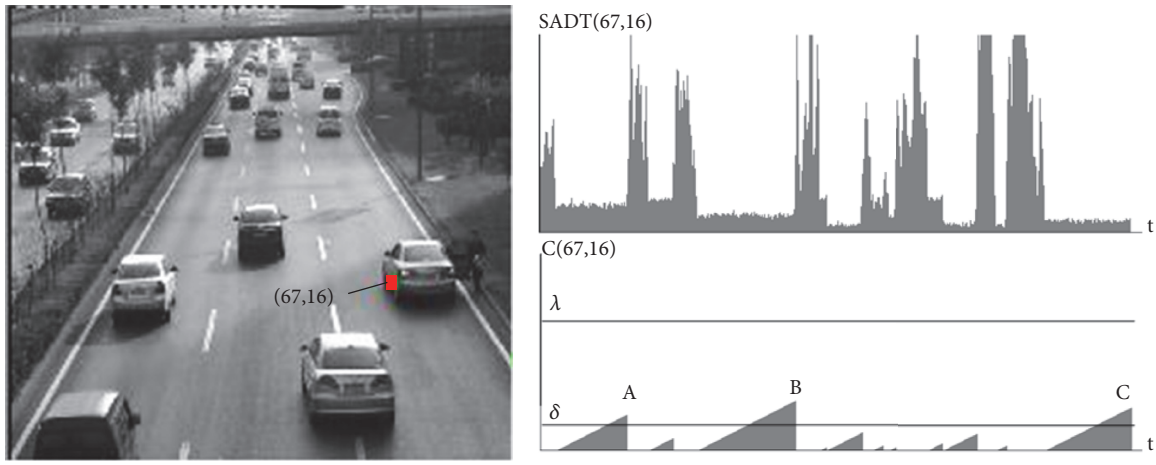
(a) Original image (b) Enlarged edge enhancement map of block (6, 30) (c) Histogram corresponding to (b)

Figure 15: Texture analysis of the false alarm block.



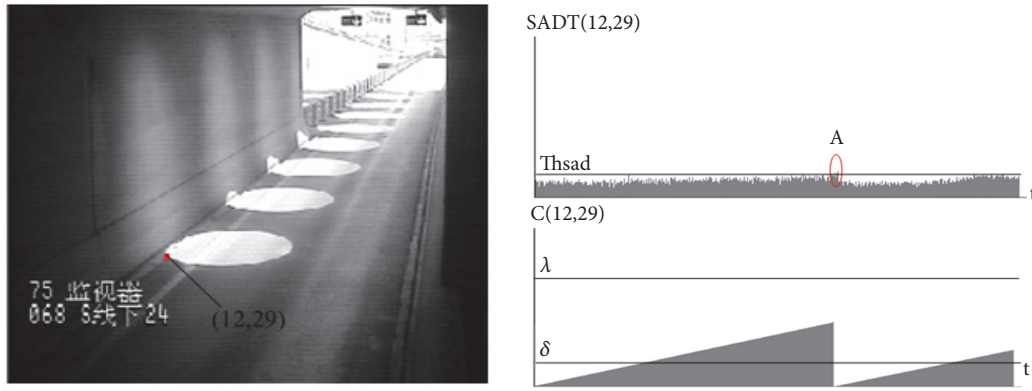
(a) Original image (b) Enlarged edge enhancement map of block (12, 51) (c) Histogram corresponding to (b)

Figure 16: Texture analysis of anomalous blocks.



(a) Original image (b) SADT and counter curve of block (67, 16) in t period

Figure 17: Effect of passing vehicles on detection.



(a) Original image (b) SADT and counter curve of block (12, 29) in t period

Figure 18: Effect of illumination on detection

3.1. Two-Way Tracking to Determine Parking and Dropping Areas. Parking and dropping objects are from the state of motion to the state of rest and have the characteristics of a forward trajectory without a backward trajectory. However, the false alarm caused by the shadow does not have such a feature, so the two-way tracking of the detected abnormal region can further reduce false alarm and improve accuracy.

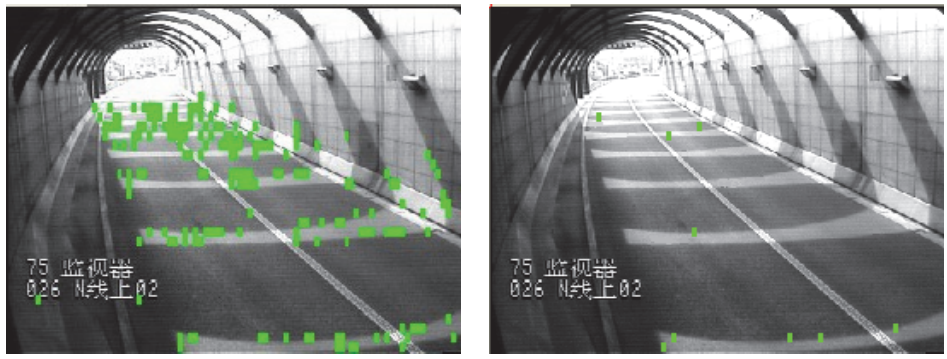
3.1.1. Corner Extraction and Corner Matching. The pixel points in the image where the brightness changes drastically and the pixel points of the maximum value of the curvature on the edge curve of the image are called corner points, which contain many pieces of important information [19]. In this paper, the *Moravec* [20] corner extraction algorithm for obtaining corner by calculating the gray-scale variance is used. This method does not depend on other local features of the object and has fast speed and real-time performance.

Figure 23 shows the results of detecting corners.

After acquiring the corner point of the abnormal block, in order to track the abnormal block, it is necessary to find the position of the corner point in the next time series to match the corner points. The matching process is the process of finding the location with the greatest similarity to a given template in the search area. Commonly used matching methods [21] include block matching method, pixel recursive method, phase correlation method, and spatial feature method. Compared with other methods, the block matching method [22] has the characteristics of small calculation and easy implementation. Therefore, the block matching method is used to obtain the motion trajectory of the abnormal block.

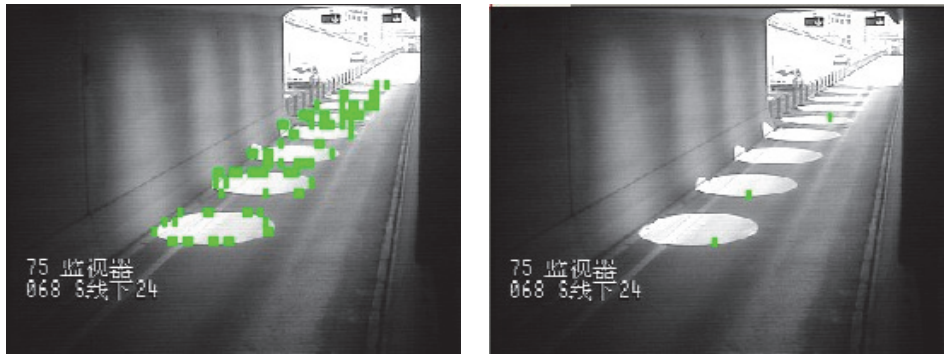
The block matching method needs to use the matching criterion to measure the matching rate between two small blocks, so it is necessary to select the matching criteria. And the matching criteria directly affect the tracking accuracy and the calculation amount. Common matching criteria [23] are mean absolute error (MAD), mean square error function (MSE), normalized cross-correlation function (NCCF), and sum of absolute error (SAD).

Among them, the MSE criterion and the NCCF criterion require more multiplication operations, the time complexity is higher, and it is rarely used. The calculation amount of SAD is smaller than that of MAD, and the implementation



(a) Before improvement (b) After improvement

Figure 19: Comparison of the effects before and after the improvement of the algorithm for the exit of Shanghai Fuxing Road Tunnel.



(a) Before improvement (b) After improvement

Figure 20: Comparison of the effects before and after the improvement of the algorithm for the entrance of Shanghai Fuxing Road Tunnel.

is simple, convenient, and widely used. Therefore, this paper uses SAD as the selection criterion for the optimal matching point.

3.1.2. Two-Way Tracking Algorithm. First, in the preliminary abnormal block, the corner point $P_0(x, y)$ is obtained by the Moravec algorithm.

Second, read the m-frame image closest to the preliminary abnormal block, and carry out backward tracking with P_0 as the initial point. Find the best matching point P_{N-1} according to the SAD criterion in the previous frame with P_0 as the initial point. Centering on P_{N-1}, continue to find the best matching point in the N-2 frame. And so on, if the target tracking trajectory Tracker (P_N, P_{N-1},... P_{N-m}) can be obtained, go to (3) for forward tracking; otherwise the exception block will not be processed.

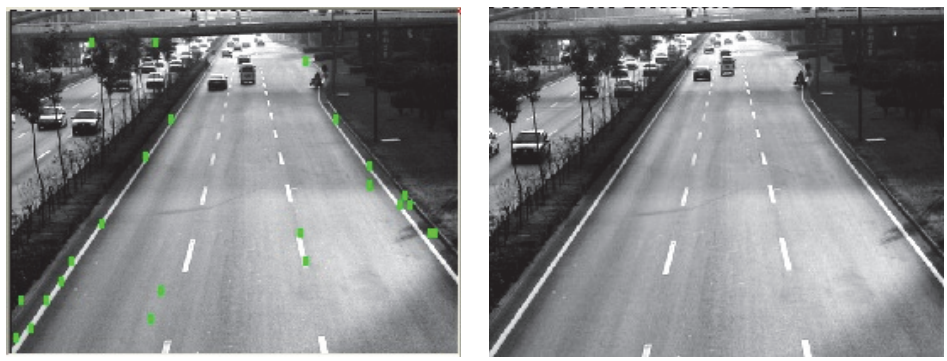
Third, forward tracking with P_0 as the initial point. Find the best matching point P_{N+1} according to the SAD criterion in the next frame with P_0 as the initial point. Focusing on P_{N+1}, continue to find the best matching point in the N+2 frame. In this way, if the target tracking trajectory Tracker (P_N, P_{N+1},...) cannot be obtained, the abnormality flag D2(m, n) of the abnormal block is set to True.

As can be seen from Figure 24, the parking or dropping object after the two-way tracking must meet the characteristics of having a backward trajectory without a forward trajectory.

3.1.3. Analysis of Two-Way Tracking Experiment Results. The comparison of the detected results before and after the two-way tracking is shown in Figure 25. It can be seen that after two-way tracking, false alarm due to shadows and the like have been completely eliminated.

3.2. Segmentation of Parking and Dropping Area Based on Connected Domain Analysis. The parking and dropping areas after two-way tracking contain many abnormal image blocks, which constitute the actual target area. Therefore, in order to further determine the position and size of the target area and segment the target area, the connected domain analysis of the image is performed in units of image blocks.

3.2.1. Common Connected Domain Analysis Algorithm. An image area that is adjacent in position and has the same pixel value is generally referred to as an image connected domain. The process of finding and marking all connected domains in an image is called connected domain analysis. There are many methods for connected domain analysis. Here are two of the most commonly used algorithms: two-pass scanning and seed filling. A method of finding all



(a) Before improvement (b) After improvement

Figure 21: Comparison of the effects before and after the improvement of the algorithm for Xi'an South Second Ring Road.

(a) Frame 786 before improvement (b) Improved frame 755

Figure 22: Comparison of the effect of detecting parking.

connected domains in an image and marking them by repeatedly scanning the images twice is called a two-pass scanning method [24]. Seed filling is often used to fill graphics. A foreground pixel is selected as a seed, and then all foreground pixels adjacent to the seed position and having the same pixel value as the seed are referred to as connected domains.

The above are two basic connected domain analysis methods. In view of the fact that the two-pass scanning method requires two scans of the image, and a large amount of space is needed to store the equal relationship between the markers, the seed filling method is used to analyze the connected region of the abnormal region.

3.2.2. Algorithm Description and Experimental Results. After the abnormal block detection, the abnormality D1(m, n) and D2(m, n) of each image block exist the following three cases:

The first one is D1(m, n)= False and D2(m, n)= False, which means that the image block has neither state change nor track feature.

The second is D1(m, n)=True and D2(m, n)=False, indicating that the image block has a state change but does not meet the track characteristics.

The third type is D1(m, n)=True and D2(m, n)=True, indicating that the image block has both state changes and trajectory characteristics.

The image block of the second case is an abnormal area determined initially, and the image block of the third case is an abnormal area caused by the occurrence of a parking or a dropping event.

(1) Algorithm Description. Firstly, scan the image in rows in units of image blocks. If an abnormal block is scanned, a new connected region is considered to appear.

(a) The current scanned exception block is seeded and marked. And the upper, lower, left and right boundaries of the connected area are set to be the boundary of the abnormal block. Then, the eight image blocks adjacent to the abnormal block are sequentially scanned, and if there are abnormal blocks, all the abnormal blocks are pushed onto the stack.

(b) Popping the top block of the stack, giving the same mark, and updating the four boundaries of the connected domain according to the positional relationship between the position of the abnormal block and the upper, lower, left and right boundaries. Then, the eight image blocks adjacent to the abnormal block are sequentially scanned. If there are abnormal blocks, all the abnormal blocks are pushed onto the stack.

(c) Repeat (b) until the stack is empty. Then a connected area with four known boundaries in the image is

(a) Corner of parking on Chongqing Expressway

(b) Corner of the dropping object of Chongqing Expressway

(c) Corner of parking in Xi'an South Second Ring Road

Figure 23: Detected corner points.

(a) Results of bidirectional tracking of Figure 23(a)

(b) Results of bidirectional tracking of Figure 23(b)

(c) Results of bidirectional tracking of Figure 23(c)

Figure 24: The result of bidirectional tracking of the corner points in Figure 24.

found, and all the exception blocks in the connected area are marked as True.

Second, repeat step one until the end of the scan.

After the scan, all connected domains in the image can be found. All the exception blocks in each connected domain have the same mark, and four boundaries of each connected domain can be derived. By considering each connected domain as a target area, you can determine the location and size of each target area, as shown in Figure 26.

(2) Analysis of Results. As can be seen from Figure 26, the abnormal region obtained by the above connected domain analysis algorithm is smaller than the actual abnormal region, and there is a case where the same abnormal target is divided into a plurality of abnormal targets, which seriously affects subsequent processing.

3.2.3. Existing Problems and Solutions. The above method of connected domain analysis has defects, mainly reflected in the following two aspects:

(1) Due to the interference of illumination, vehicles, pedestrians, etc., the timing of each image block in the abnormal target reaching an abnormal state may be different, as shown in Figure 27.

In Figure 27, A is an abnormal target, and the gray value of each block is different. The larger the gray value, the less time it takes for the block to reach the abnormal state.

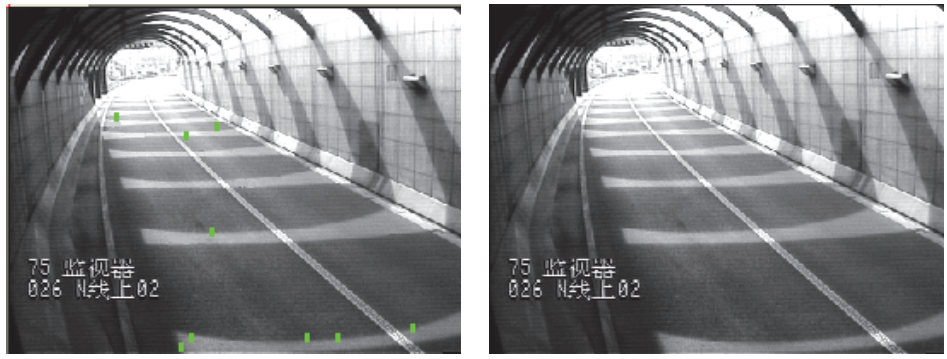
In view of this situation, it is necessary to save the abnormality flag of the detected abnormal block for a while (usually 2~3 seconds). If an image block with an abnormality flag D1(m, n)=D2(m, n)=True appears around the block during this time, the block is considered to meet the adjacent condition.

(2) When using the two-way tracking method to reduce false positives caused by shadows and other disturbances, some image blocks in the abnormal target are also removed. Thereby causing the originally connected area to become a nonconnected area, and one target becomes multiple targets. Therefore, when carrying out the connected domain analysis, all the image blocks with the abnormality flag D1(m, n)=True and adjacent to the seed are pushed onto the stack. (The initial seed must be an exception block).

3.2.4. Improved Algorithm and Experimental Results. According to the two shortcomings proposed in Section 3.2.3, the connected domain analysis method is improved. The specific steps are as follows.

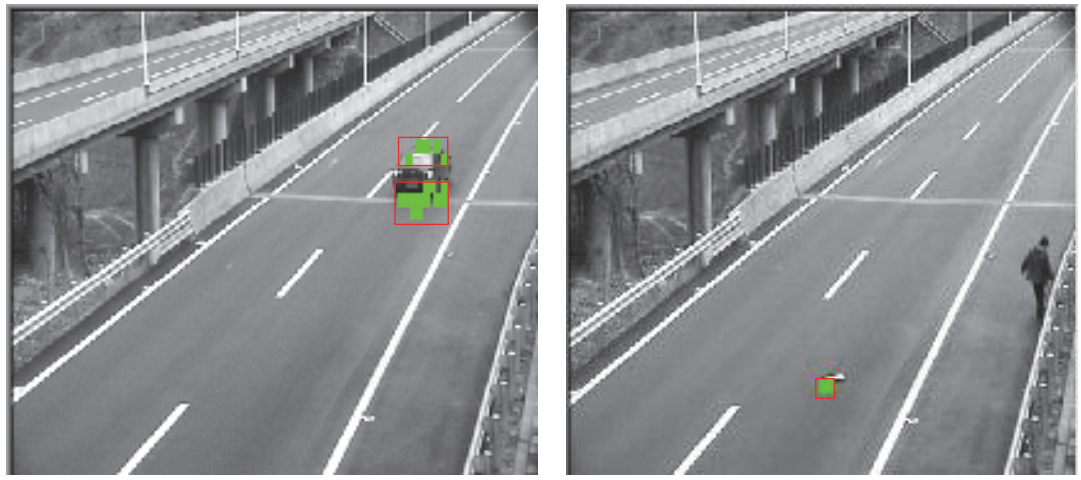
Firstly, scan the image in rows in units of image blocks. If an abnormal block is scanned, a new connected region is considered to appear.

(a) The current scanned exception block is seeded and marked. And the upper, lower, left and right boundaries of the connected area are set to be the boundary of the abnormal block. Then, the eight image



(a) Before two-way tracking (b) After two-way tracking

Figure 25: Comparison of abnormal blocks detected before and after bidirectional tracking.



(a) Parking (b) Dropping

Figure 26: Results of connected domain analysis.

blocks adjacent to the abnormal block are sequentially scanned, and If there are image blocks with abnormal markers D1(m, n)=True in these eight image blocks, all the blocks are pushed onto the stack.

(b) Popping the top block of the stack, giving the same mark, and updating the four boundaries of the connected domain according to the positional relationship between the position of the abnormal block and the upper, lower, left and right boundaries. Then, the eight image blocks adjacent to the abnormal block are sequentially scanned. If there are image blocks with abnormal markers D1(m, n)=True in these eight image blocks, all the blocks are pushed onto the stack.

(c) Repeat (b) until the stack is empty. Then a connected area with four known boundaries in the image is found, and all the exception blocks in the connected area are marked as True.

Secondly, Repeat step one until the end of the scan.

After the scan is finished, all the connected domains in the image can be acquired; likewise, each connected domain is considered as a target region, and the experimental result is shown in Figure 28.

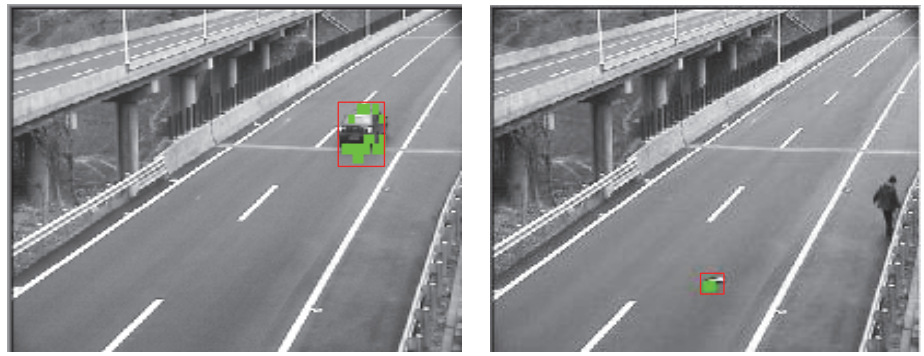
By comparing Figure 26 with Figure 28, it can be found that the anomalous region obtained by the improved connected domain analysis method is more accurate.

4. Separation of Parking and Dropping Objects

The work done in the previous section only defines the parking and dropping area in the image and does not distinguish the target type, that is, whether it is parking or dropping. The traditional methods of differentiation like area, rate of change of the direction of motion, and average speed have limitations. This section describes how to differentiate between targets using 3D information.

4.1. Calibration. The use of a camera to capture images or video will result in the loss of target information, which will cause a series of problems such as geometric deformation and scale change when processing images in traffic scenes. If you can use the image to restore objects in the space and

Figure 27: Status difference of different blocks in the abnormal area.



(a) Parking (b) Dropping

Figure 28: Improved results.

use the properties of the object itself, these problems will be completely eliminated. In such an environment, camera calibration technology was born. The main purpose of this technology is to define the internal and external parameters of the camera under a specific imaging model and then establish the relationship between image pixel coordinates and world coordinates.

The camera imaging process can be described by its imaging model. In this paper, we use a linear imaging model for calibration, which can make calculations easier.

With the deepening of calibration technology research, many scholars have proposed a representative calibration method [25]. In 1981, Martins [26] proposed an algorithm for camera calibration using two planes. This method simplifies the process of solving based on meeting the calibration accuracy. In 1986, R.Y.Tsai [27] established the Tsai camera model and proposed a two-step calibration method. This method simplifies the process of solving on the basis of meeting the calibration accuracy. In 1998, Zhang Zhengyou [28] proposed a method for camera calibration using 2D planar targets. This method only needs to make a calibration plate and then shoot the calibration plate at different angles. By detecting the feature corners in each photo, you can estimate the camera's internal and external parameters. The method has the advantages of low cost, high precision, and the like, and has been widely applied. In 2014, Zheng Yuan [29] proposed a camera calibration method based on vanishing point. Its principle is as follows: as long as the vanishing point formed in the image by the parallel lines in three directions in the space, and the height of the camera in the space or a known distance on the road surface or a height perpendicular to the road surface, you can find the internal and external parameters of the camera.

Consider the existence of a large number of parallel markings in traffic scenes, and the country has uniform standards for the dimensions of these markings. Therefore, the use of these marking lines can not only easily find the vanishing point in three directions, but also easily find a line segment of a known distance on the image. Therefore, in the paper, employ the method of camera calibration based on vanishing point proposed by Zheng Yuan et al.

Since the Direct Linear Transformation (DLT) [30] requires six known points as inputs, the determination of these six points requires special markers [31]. If such a marker cannot be found in the scene to be calibrated [32], the calibration cannot be completed. No known points are used as input, and calibration can be done in scenes where the DLT cannot be calibrated using only some of the markings on the road surface and known camera parameters. Calibration is usually performed using the vanishing point in the image.

4.2. Parking and Dropping Object Distinguishing Method Based on Image Inverse Perspective Mapping. The image captured by the camera is a three-space to two-dimensional

(a) Van (b) Car (c) Suv

Figure 29: A rough three-dimensional model frame.

projection, mathematically known as the perspective mapping. According to the camera imaging model, the closer to the camera, the larger the object. And the image during projection has been deformed and physical size information has been lost.

In this paper, the inverse perspective mapping method [33] is used to establish the mapping relationship between the two-dimensional image plane and the three-dimensional space, so as to eliminate the deformation and recover the physical size of the object [34]. Points in 3D space mapped to image are unique, but there are many possibilities for inverse perspective mapping from 2D images to 3D space [35]. Points in one image correspond to points in multiple 3D spaces.

If a plane is determined in advance in three-dimensional space, the points on the plane are one-to-one correspondence with those on the two-dimensional image, and the plane is called a back-projection plane. In this way, the data in the two-dimensional image can be mapped onto the back-projection plane to obtain a map, which contains the information of the back-projection plane. And the closer the back-projection plane is to the surface of the object [36], the more accurate the three-dimensional information of the object is expressed.

We set the back-projection surface in the anomalous area and get the true size of the abnormal target through the corresponding back-projection map. If the size is close to the size of the real vehicle [37], the target is considered to be parking; otherwise it is a dropping object.

4.2.1. Algorithm Implementation

Step 1. In the abnormal pixel area, the circumscribed rectangular area of the vehicle in the image is located, and the three-dimensional model frame of the detected vehicle is roughly determined according to the circumscribed rectangle, as shown in Figure 29.

Step 2. Extractan edge of the circumscribed rectangular area of the vehicle, and construct the inverse projection of the bottom surface of the three-dimensional model frame on the obtained edge map. On the edge map, the linear segment extraction method proposed in this paper (in Section 4.2.3) is employed to extract the edge lines of the front and side of the vehicle chassis, and record the intersection point P of the two lines. The extraction results are shown in Figure 30.

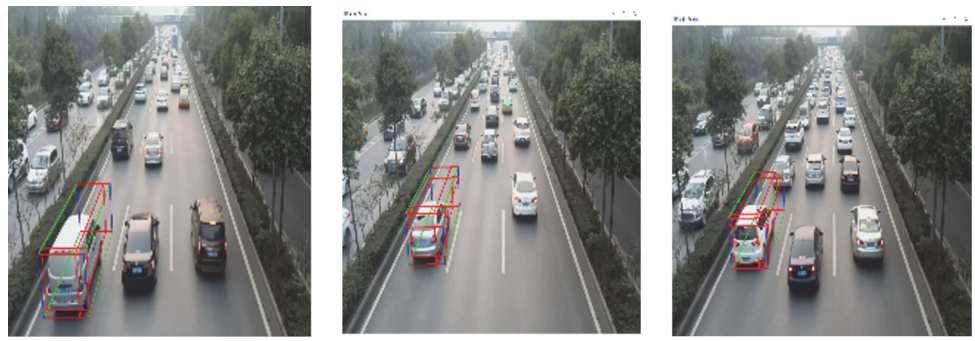
Figure 30: Extraction results of the chassis line.

Step 3. The area is defined by the plane that contains the bottom line (the red one in Figures 31 and 32) and is perpendicular to the road surface and the intersection of the three-dimensional model frame, which is the inverse projection of the front of the vehicle. Then according to the principle of the inverse projection, the reverse projection image can be defined as shown in Figures 33, 34, 37, and 38. In the same way, as shown in Figures 35 and 36, it is the left view.

Step 4. The line extraction algorithm described in Section 4.2.3 is used for the left view and the front view obtained in the Step 3, and the results are shown in Figures 39–44. Since the back-projected image can reflect the actual physical size of the object, the actual height of the vehicle can be obtained directly from the topmost line in the front view. And assume that the height of the vehicle is H.

Step 5. By the intersection P defined in Step 2 and the vehicle height H defined in Step 4, the position of the reverse projection top view can be determined, as shown in Figures 37, 38, and 45. Similarly, the result of the straight line detection of the top view can be obtained, as shown in Figures 43, 44, and 46.

Step 6. The three-dimensional information of the target is defined based on the result of the line extraction on the three



(a) Van (b) Car (c) Suv

Figure 31: Location of the back-projection plane of the front and left views.

Figure 32: Location of the back-projection plane of the front and left views of truck.

Figure 33: The inverse projection of the main view of van.

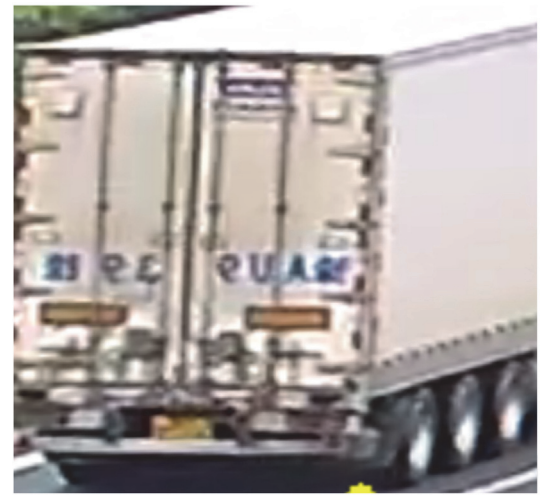


Figure 34: The inverse projection of the main view of truck.

Figure 35: The inverse projection of the left view of van.

views. The straight line distance between the left and right sides of the top view defines the width of the object. The distance between the leftmost and rightmost of all lines in the left view defines the length of the object. The height of the object is defined from the main view in Step 4. The result of the detection of the final three-dimensional size of the white vehicle is shown in Table 1.

Table 1 is the estimation process of the three-dimensional size of the vehicle target. If any of the above steps are not performed correctly, then the abnormal area is considered to be dropping rather than parking [38–40]. For example, the number of straight lines detected on the plane of the reverse projection is too small or the target does not have a chassis edge. If all the steps are completed, then it can be determined whether the target is a vehicle target or a throw according to the three-dimensional information.

4.2.2. Construction Method of Reverse Projection. There are three inverse projective planes involved in this paper, which are the tail (or head), side and top of the vehicle, as shown in Figure 48. The inverse projective plane is divided into a grid according to a certain resolution. Since the position of the inverse projective plane has been determined, then the adverse projection relationship between the projection area and the adverse projection area in the image is determined, a one-to-one mapping relationship between coordinates of the three-dimensional space and the pixel coordinates of the image can be established [41]. The specific construction process is mapping each pixel point in the projection area

Table 1: Test results for white vehicle size.

	Object width	Object length	Object height
Van	188cm	405cm	197cm
Car	175cm	405cm	147cm
SUV	169cm	390cm	168cm
Truck	254cm	1843cm	321cm

Figure 36: The inverse projection of the left view of truck.

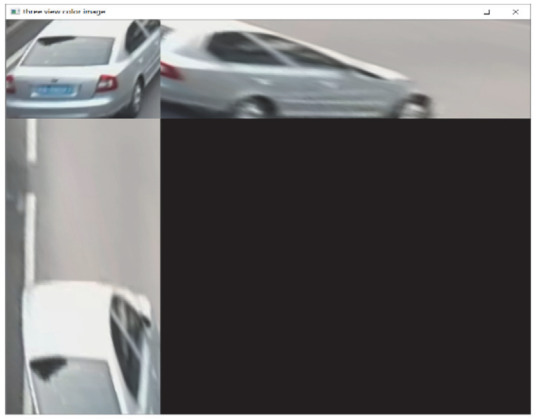


Figure 37: Three-view color image of car.

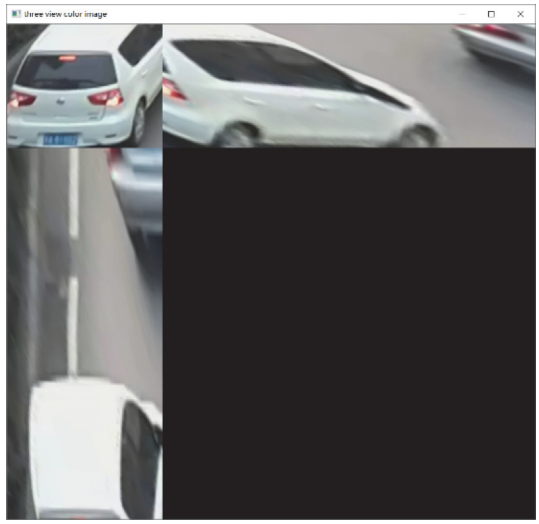


Figure 38: Three-view color image of suv.

Figure 39: Extraction result of the line in the main view of van.

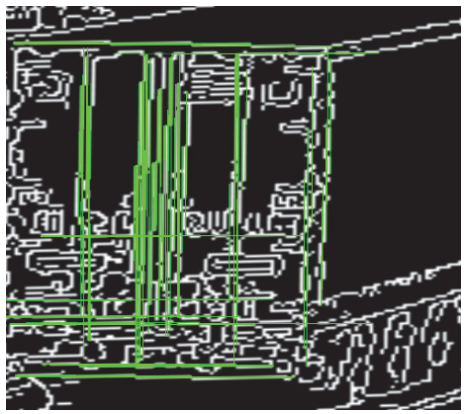


Figure 40: Extraction result of the line in the main view of truck.

on the image to a corresponding pixel point in the reverse projection image, that is, copying information in each small grid on the inverse projective plane into the reverse projection image. As shown in Figure 48, m denotes a small grid in the inverse projection plane, p denotes that m maps to a pixel point on the image, and mp denotes a pixel point corresponding to the grid m on the reverse projection image. Therefore, the inverse projection is the process of mapping pixel points in an image to pixel points of a reverse projection image.

As can be seen from the Figure 48, an inverse projective plane is constructed close to the target surface, and the corresponding reverse projection image is a copy of the target surface. The problem of geometric deformation of the target surface in the image due to camera perspective is eliminated [42], and the true characteristics of the target surface are well reflected, and the actual physical size of the object in the image can be reflected as realistically as possible.

4.2.3. A Method for Detecting Straight Line Based on the Reverse Projection Image. In this application, it is necessary to detect the linear segments of the vehicle body. And the smooth design of the modern vehicle manufacturing process makes the original distinct straight line segment on the

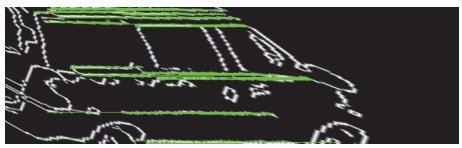


Figure 41: Extraction result of the line in the left view of van.

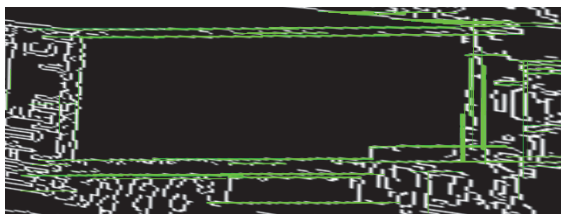


Figure 42: Extraction result of the line in the left view of truck.

Figure 43: Extraction result of the line of car.

contour of the vehicle smooth and inconspicuous. Therefore, the conventional method of extracting a straight line cannot link the partial broken edge and the curved segments with less curvature, and the detected straight line is prone to breakage.

This paper designs a method of edge coding based on the reverse projection image. If it is the edge of the image, it is encoded as 1, not the edge of the image is encoded as -1. Calculate and get the line segment with the largest sum of the directions in which the extracted line is located. An example of the encoding of an edge is shown in Figure 47 .

The advantage of this is that, without the use of time-consuming and complex algorithms such as Hough transform, the straight line is corrected using a back-projection image, so that the algorithm for detecting the line only needs to consider vertical or horizontal lines. The complexity of the algorithm is reduced and the accuracy of the detected line can be greatly improved, and the line with small curvature and local fracture is well inclusive. The algorithm flow is shown in Figure 49.

4.3. Experimental Results and Analysis. The algorithm proposed in the paper is implemented in VC6.0 environment [43], and the effectiveness of the algorithm is verified under

Figure 44: Extraction result of the line of suv.

Figure 45: Inverse projection of the top view of van.

Figure 46: Extraction result of the line in the top view of van.

−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1
1	1	−1	−1	−1	−1	−1	−1	−1	−1	−1	1	1
−1	−1	1	1	1	−1	1	1	1	−1	1	−1	−1
−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1	−1

Figure 47: Edge coding.

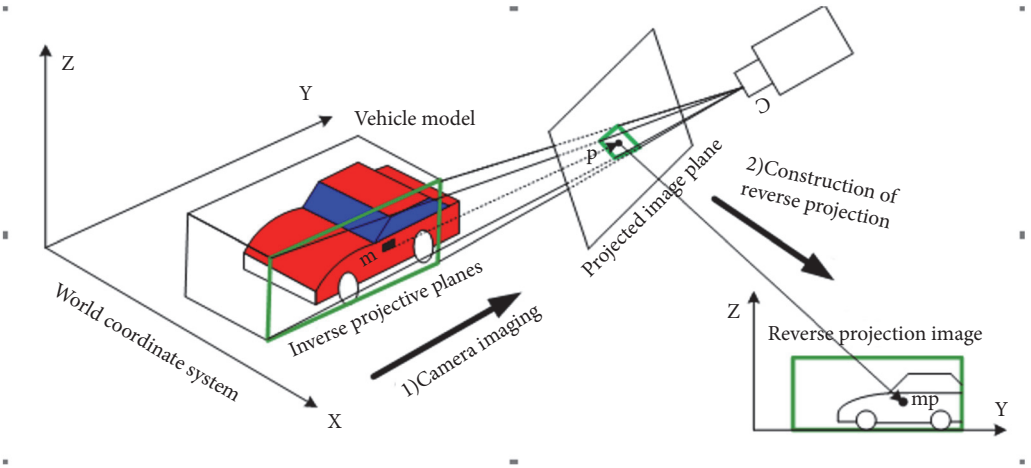


Figure 48: Construction of the inverse projection.

Table 2: Parking detection results in different scenarios.

scenarios	Number of parking events	Total number of alarms	The total number of correct alarms	Recognition rate %	Missed detection rate %	False detection rate %	Average time required for testing /ms
Xi'an South Second Ring	38	39	37	97.3	2.63	5.12	4300
Beijing Yanqing Road Section	52	54	49	94.2	5.77	9.25	4900
Chongqing Expressway	43	45	42	97.7	2.32	4.44	4100
Shanghai Outer Ring Road	60	63	58	96.7	3.33	7.93	4800
Shanghai Fuxing Road Tunnel	26	28	25	96.1	3.84	10.0	5000

different traffic scenarios. The algorithm proposed in the paper is implemented in VC6.0 environment, and the effectiveness of the algorithm is verified under different traffic scenarios. Due to the limited length of the article, five representative scenes are taken as examples, including Xi'an South Second Ring Road, Shanghai Fuxing Road Tunnel, Shanghai Overseas Ring Road, Beijing Yanqing Road Section, and Chongqing Expressway. Collect real-time video in these scenarios and use the algorithm [44, 45] designed in this paper to detect parking and dropping events. The test results are shown in Tables 2 and 3. The results of the parking test are shown in Table 2. The results of the dropping test are shown in Table 3.

It can be seen from Tables 2 and 3 that the algorithm can detect parking and dropping events more accurately in poor quality video scenes such as urban roads with high traffic flow and high-speed roads or tunnels with fast traffic speeds. Among them, the recognition rate of parking can reach more than 94%, the recognition rate of dropping objects can reach more than 92%, and the abnormality can be alarmed within 5s. Moreover, the missed detection rate and false detection rate of the parking event can be controlled below 10%, the missed detection rate of the dropping can be controlled below 10%, and the false detection rate is controlled below 20%. Therefore, the algorithm proposed in the paper has better detection accuracy [46].

5. Conclusion

The existing algorithms for detecting parking and dropping objects generally have two shortcomings: significant dependence on background and inaccurate distinction between parking and dropping objects.

In view of the above deficiencies, we study on two aspects: the detection of stationary targets and the differentiation of target types. First, based on the method of the status change, when there is an object that did not exist before in the traffic scene, the abnormal region in the scene is preliminarily defined, and then the initially determined abnormal region is bidirectionally tracked to further determine the target from the motion to the stationary in the scene. Finally, the eight-neighbor seed filling method is used to segment the target area. Therefore, the dependence on the background is reduced. Only tracking the target area where the state changes is needed, which significantly reduces the amount of computation. Second, a method of using the three-dimensional information of the target to distinguish the target type is proposed. Firstly, use the difference between

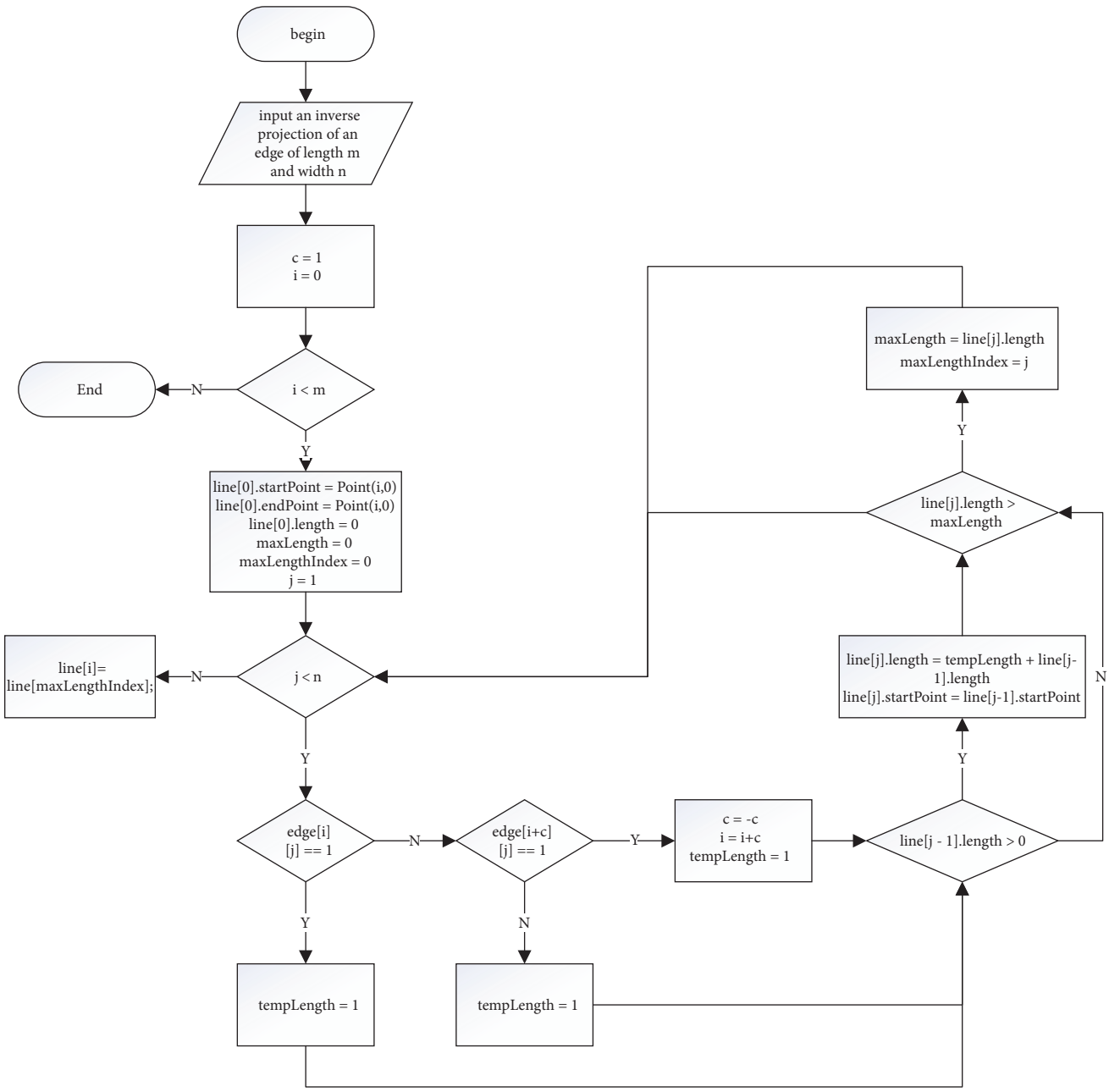


Figure 49: Algorithm for detecting straight lines based on the inverse projection.

the projections of the feature points to determine the relative height between the feature points, and use the height to distinguish the parking and dropping objects. Secondly, establish a 3D wireframe model of common vehicle model. The parking and the dropping objects are distinguished by the matching of the projection of the wireframe model on the two-dimensional image and the area of the target area. Thirdly, by establishing the inverse projection planes of different heights, the three-dimensional information of the length, width, and height of the target is obtained, and the parking and the dropping objects are distinguished by the size of the vehicle which is known. This method of distinguishing target types using three-dimensional information can not only accurately distinguish the parking and dropping objects, but also roughly classify the models of stationary vehicles.

By testing in a large number of different traffic scenarios [47], the results show that the algorithm can effectively detect the parking and dropping objects based on the low miss detection rate and false detection rate and can meet the real-time requirements.

Data Availability

The measured data used to support the findings of this study have not been made available because it belongs to the local

Table 3: Detection results of the falling objects in different scenarios.

scenarios	Number of parking events	Total number of alarms	The total number of correct alarms	Recognition rate %	Missed detection rate %	False detection rate %	Average time required for testing /ms
Xi'an South Second Ring	14	15	13	92.8	9.09	13.3	4300
Beijing Yanqing Road Section	19	22	18	94.7	7.14	18.2	4900
Chongqing Expressway	16	18	15	93.7	6.25	16.7	4100
Shanghai Outer Ring Road	15	17	14	93.3	6.67	17.6	4800
Shanghai Fuxing Road Tunnel	13	14	12	92.3	7.69	14.3	5000

authorities of traffic control and management in Shanghai, Xi'an, and Chongqing, China.

Conflicts of Interest

The authors declare that all data availability in this article are true and reliable and there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The work was funded by the Project of Shaanxi Provincial Science and Technology Program (Grant no. 2014JM8351), the Fundamental Research Funds for the Central Universities (Grants nos. 2013G1241109 and 300102248305), and the National Natural Science Foundation of China (Grant no. 61501058). Thanks are due to Liting Sun for the great work she has done.

References

[1] M. Alam, D. Moroni, G. Pieri et al., "Real-time smart parking systems integration in distributed ITS for smart cities," *Journal of Advanced Transportation*, vol. 2018, Article ID 1485652, 13 pages, 2018.

[2] J. Li, C. Liu, J. X. Yu, Y. Chen, T. Sellis, and J. S. Culpepper, "Personalized influential topic search via social network summarization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 7, pp. 1820–1834, 2016.

[3] A. Bevilacqua and S. Vaccari, "Real time detection of stopped vehicles in traffic scenes," in *Proceedings of the 2007 IEEE Conference on Advanced Video and Signal Based Surveillance*, pp. 266–270, DBLP, London, UK, September 2007.

[4] B.-F. Wu, C.-C. Kao, C.-C. Liu, C.-J. Fan, and C.-J. Chen, "The vision-based vehicle detection and incident detection system in Hsueh-Shan tunnel," in *Proceedings of the 2008 IEEE International Symposium on Industrial Electronics, ISIE 2008*, pp. 1394–1399, UK, July 2008.

[5] S. Guler, J. A. Silverstein, and I. H. Pushee, "Stationary objects in multiple object tracking," in *Proceedings of the 2007 IEEE Conference on Advanced Video and Signal Based Surveillance, AVSS 2007*, pp. 248–253, UK, September 2007.

[6] Z. Beibei and W. Qisheng, "The study of highway illegal parking real-time detection algorithm based on video," *Journal of Electronic Science and Technology*, vol. 24, no. 9, pp. 20–23, 2011.

[7] R. Akhawaji, M. Sedky, and A.-H. Soliman, "Illegal parking detection using Gaussian mixture model and kalman filter," in *Proceedings of the 14th IEEE/ACS International Conference on Computer Systems and Applications, AICCSA 2017*, pp. 840–847, Hammamet, Tunisia, November 2017.

[8] H. Tianfu, B. Jie, L. Ruiyuan et al., "Detecting vehicle illegal parking events using sharing bikes' trajectories," in *roceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 340–349, London, UK, August, 2018.

[9] F. Porikli, "Detection of temporarily static regions by processing video at different frame rates," in *Proceedings of the 2007 IEEE Conference on Advanced Video and Signal Based Surveillance, AVSS 2007*, pp. 236–241, UK, September 2007.

[10] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, 2000.

[11] D.-H. Wang, H.-Y. Hu, Z.-H. Li, and Z.-W. Qu, "Detection and recognition algorithm of illegal parking," *Journal of Jilin University (Engineering and Technology Edition)*, vol. 40, no. 1, pp. 42–46, 2010.

[12] M. Chunyang, M. Xing, and Z. Panpan, "Smart detection of vehicle in illegal parking area by fusing of multi-features," in *Proceedings of the 9th International Conference on Next Generation Mobile Applications, Services and Technologies, NGMAST 2015*, pp. 388–392, UK, September 2015.

[13] X. Xuemei, W. Chenye, S. Chen, S. Guangming, and Z. Zhao, "Real-time illegal parking detection system based on deep learning," in *Proceedings of the 2017 International Conference on Deep Learning Technologies (ICDLT '17)*, pp. 23–27, ACM, New York, NY, USA, 2017.

[14] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering - ASME Journal of Fluids Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[15] M.-H. Lin, J. G. Carlsson, D. Ge, J. Shi, and J.-F. Tsai, "A review of piecewise linearization methods," *Mathematical Problems in Engineering*, vol. 2013, Article ID 101376, 8 pages, 2013.

[16] X. Su, W. Fang, Q. Shen, and X. Hao, "An image enhancement method using the quantum-behaved particle swarm optimization with an adaptive strategy," *Mathematical Problems in Engineering*, vol. 2013, Article ID 824787, 14 pages, 2013.

[17] S. Singh, S. Saurav, R. Saini et al., "Comprehensive review and comparative analysis of hardware architectures for sobel edge detector," *ISRN Electronics*, vol. 2014, Article ID 857912, 9 pages, 2014.

[18] L. Yu, "Image noise preprocessing of interactive projection system based on switching filtering scheme," *Complexity*, vol. 2018, Article ID 1258306, 10 pages, 2018.

[19] R.-Y. Tang and Z.-M. Zeng, "Adaptive fractional differentiation harris corner detection algorithm for vision measurement of surface roughness," *Advances in Mathematical Physics*, vol. 2014, Article ID 494237, 6 pages, 2014.

[20] J. Khan, S. Bhuiyan, and R. Adhami, "Feature point extraction from the local frequency map of an image," *Journal of Electrical and Computer Engineering*, vol. 2012, Article ID 182309, 15 pages, 2012.

[21] Q. Peng, L. Tu, K. Zhang, and S. Zhong, "Automated 3D scenes reconstruction using multiple stereo pairs from portable four-camera photographic measurement system," *International Journal of Optics*, vol. 2015, Article ID 471681, 9 pages, 2015.

[22] S. Zhu, L. Li, J. Chen, and K. Belloulata, "An efficient fractal video sequences codec with multiviews," *Mathematical Problems in Engineering*, vol. 2013, Article ID 853283, 8 pages, 2013.

[23] C. Choi and J. Jeong, "Fast motion estimation algorithm using dual bit-plane matching criteria," in *Proceedings of the 2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2014*, pp. 398–401, Croatia, May 2014.

[24] W. Wu, Z. Zhou, S. Wu, and Y. Zhang, "Automatic liver segmentation on volumetric CT images using supervoxel-based graph cuts," *Computational and Mathematical Methods in Medicine*, vol. 2016, Article ID 9093721, 14 pages, 2016.

[25] L. Wang, S. Lin, J. Yang et al., "Dynamic traffic congestion simulation and dissipation control based on traffic flow theory model and neural network data calibration algorithm," *Complexity*, vol. 2017, Article ID 5067145, 11 pages, 2017.

[26] H. A. Martins, J. R. Birk, and R. B. Kelley, "Camera models based on data from two calibration planes," *Computer Graphics and Image Processing*, vol. 17, no. 2, pp. 173–180, 1981.

[27] R. Y. Tsai, "A versatile camera calibration technique for high accuracy. 3D machine vision metrology using off-the-shelf TV cameras and lenses," *IEEE Transactions on Robotics and Automation*, vol. 3, no. 4, pp. 323–344, 1987.

[28] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[29] Y. Zheng and S. Peng, "A practical roadside camera calibration method based on least squares optimization," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 15, pp. 831–843, 2014.

[30] Y. I. Abdel-Aziz, H. M. Karara, and M. Hauck, "Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry," *Photogrammetric Engineering and Remote Sensing*, vol. 81, no. 2, pp. 103–107, 2015.

[31] L. Jianxin, W. Xinjue, D. Ke et al., "Discovering influential community over large social networks," in *Proceedings of the 33rd IEEE International Conference on Data Engineering (ICDE)*, 2017.

[32] S. D. Ma, "A self-calibration technique for active vision systems," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 1, pp. 114–120, 1996.

[33] H. Li, M. Feng, and X. Wang, "Inverse perspective mapping based urban road markings detection," in *Proceedings of the 2nd IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS '12)*, pp. 1178–1182, Hangzhou, China, November 2012.

[34] J. Li, C. Liu, and J. X. Yu, "Context-based XML keyword query diversification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 3, pp. 660–672, 2015.

[35] W. Wang, C. Shen, J. Zhang, and S. Paisitkriangkrai, "A two-layer night-time vehicle detector," in *Proceedings of the Digital Image Computing: Techniques and Applications (DICTA '09)*, pp. 162–167, Melbourne, Australia, December 2009.

[36] J. Li, C. Liu, R. Zhou, and J. X. Yu, "Quasi-SLCA based keyword query processing over probabilistic XML data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 4, pp. 957–969, 2014.

[37] A. Neubeck and L. Van Gool, "Efficient non-maximum suppression," in *Proceedings of the 18th International Conference on Pattern Recognition (ICPR '06)*, pp. 850–855, Hong Kong, China, August 2006.

[38] Y. Ke, F. Qin, W. Min, and G. Zhang, "Exposing image forgery by detecting consistency of shadow," *The Scientific World Journal*, vol. 2014, Article ID 364501, 9 pages, 2014.

[39] S. Y. Chen, H. Tong, and C. Cattani, "Markov models for image labeling," *Mathematical Problems in Engineering*, vol. 2012, Article ID 814356, 18 pages, 2012.

[40] N. Zhou, T. Yang, and S. Zhang, "An improved FCM medical image segmentation algorithm based on MMTD," *Computational and Mathematical Methods in Medicine*, vol. 2014, Article ID 690349, 8 pages, 2014.

[41] B. Wang, T. Su, X. Jin, J. Kong, and Y. Bai, "3D reconstruction of pedestrian trajectory with moving direction learning and optimal gait recognition," *Complexity*, vol. 2018, Article ID 8735846, 10 pages, 2018.

[42] L. Zhao, L. F. Liu, Q. Y. Wang, T. J. Li, and J. H. Zhou, "Moving target detection and active tracking with a multicamera network," *Discrete Dynamics in Nature and Society*, vol. 2014, Article ID 976574, 11 pages, 2014.

[43] K. F. Ackermann, B. Hoffmann, L. S. Indrusiak, and M. Glesner, "Providing memory management abstraction for self-reconfigurable video processing platforms," *International Journal of Reconfigurable Computing*, vol. 2009, Article ID 851613, 15 pages, 2009.

[44] M. Roca-Riu, J. Cao, I. Dakic, and M. Menendez, "Designing dynamic delivery parking spots in urban areas to reduce traffic disruptions," *Journal of Advanced Transportation*, vol. 2017, Article ID 6296720, 15 pages, 2017.

[45] B. Tian, Y. Li, B. Li, and D. Wen, "Rear-view vehicle detection and tracking by combining multiple parts for complex urban surveillance," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 2, pp. 597–606, 2014.

[46] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, Mateo, CA, USA, 1988.

[47] J. Zhao, M. Han, C. Li, and X. Xin, "Visibility video detection with dark channel prior on highway," *Mathematical Problems in Engineering*, vol. 2016, Article ID 7638985, 21 pages, 2016.

Hindawi
Complexity
Volume 2019, Article ID 9871971, 16 pages
https://doi.org/10.1155/2019/9871971

Research Article

Location-Aware Web Service Composition Based on the Mixture Rank of Web Services and Web Service Requests

Junwen Lu,[1] Guanfeng Liu ,[2] Keshou Wu,[1] and Wenjiang Qin[3]

[1]*Engineering Research Center for Software Testing and Evaluation of Fujian Province, Xiamen University of Technology, Xiamen, China*
[2]*Department of Computing, Macquarie University, Sydney, NSW, Australia*
[3]*Petro China Northwest Sales Company, China*

Correspondence should be addressed to Guanfeng Liu; guanfeng.liu@mq.edu.au

Received 24 January 2019; Accepted 20 March 2019; Published 7 April 2019

Guest Editor: Jianxin Li

Web service composition is widely used to extend the function of web services. Different users have different requirements of QoSs (Quality of Services) making them face many problems. The requirement of a special QoS may be a hard requirement or a soft requirement. The hard requirement refers to the QoS which must be satisfied to the user, and the soft one means that the requirement is flexible. This paper tries to solve the service composition problem when there are two kinds of requirements of QoSs. To satisfy various kinds of requirement of the QoS, we propose a composition method based on our proposed framework. We give an analysis from composition models of services and from related QoE (Quality of Experience) of web services. Then, we rank the service candidates and the service requests together. Based on the ranking, a heuristics is proposed for service selection and composition-GLLB (global largest number of service requests first, local best fit service candidate first), which uses "lost value" in the scheduling to denote the QoE. Comparisons are used to evaluate our method. Comparisons show that GLLB reduces the value of *NUR* (Number of Unfinished service Requests), *FV* (Failure Value), and *AFV* (Average Failure Value).

1. Introduction

In recent years, high-performance hardware and software resources make the Cloud widely used by companies and departments. Cloud computing is based on Visualization and Service-Oriented Architecture (SOA). The goal of Cloud computing is to optimize the usage of physical and software resources, improve flexibility, and automate management [1]. When web services come into Cloud, they enhance services by service composition. The QoSs of the web service decides whether the Cloud can satisfy the requirement of the user. Typically, QoS requirements include execution time, scalability, high availability, trust, security, and so on [2].

A successful web service in service-oriented Cloud computing, not only tries to provide functionality as request, but also ensures to satisfy the requirement of QoSs [3]. Different users have diverse requirements to various QoSs [4]. For example, when users download files from the Internet, they always prefer to a web service with higher bandwidth; but when users transform money between banks, security is the most important QoS.

A web service only provides a single functionality for user. Web service composition enhances the ability of the Cloud, which combines multiple web services together to form a new functionality. At the same time, it also brings problems for the scheduling of web services [5]. Quality of Experience (QoE) is a measure of the level of customer satisfaction of a service provider [6]. While QoS is to illustrate the quality of services, which contains several attributes of the services. When users have many service composition requests, the selection order of web services influences the scheduling result of others, which have diverse value in QoE.

A. Jula et al. summarize nine targets for the web service selection [7], including (1) user requirement satisfaction; (2) qualitative to quantitative transformation of QoS parameters; (3) algorithm improvements; (4) introducing data storage and indexing structures; (5) self-adaptability, automaticity, increasing reliability, and accuracy, and quality assurance; (6)

proposing an improvised QoS mathematical model; (7) revenue maximization; (8) optimization of the service discovery process; (9) proposing new frameworks and structures. Most of time, those nine targets are influenced by each other, such that (3) algorithm improvements always influence (5) self-adaptability. In this paper, we try to consider multiple aspects, especially for (1), (2), (3), and (8).

When some requirements of QoS attributes are flexible (soft), the service composition problem becomes more difficult [8]. Because the flexible requirement of QoS attributes makes the selection of web service more widely, and it is difficult to decide which web service is the right selection. For example, if we assign a web service request with a web service which has a higher value in QoS attribute than the requirement, it may make others web service requests which need higher values in QoS attribute cannot be satisfied. If we assign a web service request with a web service which just satisfies the lowest requirement, the system may have low performance for the whole system, and the service may have lower value in QoE, especially when there are enough web services. The main problem of this paper is to schedule the web services when (1) the web service has different QoS attributes; (2) the service composition request has various requirements to diverse QoS attributes (hard or soft); (3) there are different functions for QoE to QoS for various kinds of requirements.

The main contributions of the paper are as follows: (1) we give a framework of web service composition; (2) we discuss web service composition methods from the QoE; (3) a new ranking of web services and requests is proposed; (4) a heuristic is proposed for web service selection and composition.

The rest of the paper is organized as follows. In the next section, we give an overview of web services from various aspects, such as the attributes, the preference of users, and web service composition methods. Section 3 describes the framework of web service composition. Section 4 discusses the service composition from the QoE. Section 5 gives a new ranking method for web services and requests. At the same time, the section also proposes a web service selection and a composition heuristic. Section 6 explains the simulation environment and the simulation results. Section 7 presents the conclusion and the future work.

2. Related Work

The difficulty of web service composition comes from many aspects. Those reasons are as follows:

(1) There are many kinds of QoS attributes about the web services [8, 9]. Those QoS attributes include execution time, reliability, availability, reputation, price, trust degree, and so on. Even a QoS attribute consists of multiple subattributes [2].

(2) There are many structures for the service composition request; the basis structures include sequence structure, parallel structure, loop structure, and case structure. And for a real service composition request, it is always a mixture of those structures that makes the service composition problem more difficult.

(3) The value of QoS attribute also has different kinds and it is difficult to get the accuracy value. The value may belong to Boolean, numerical, or even a scope. Some QoS attributes belong to objective and others belong to subjective. For the objective value, such as reputation, it is difficult to get the accurate value of the related QoS attributes.

(4) Some QoS attributes influence each other [10]. Service-dependent QoSs can be partial and full dependencies. One example is the execution time which always influences the reputation.

(5) Different users have diverse preferences to QoS attributes [4, 11–13]. Because of various targets of different users, users have diverse preferences to the QoS. In other words, even for the same QoS attributes, different users have unlike QoEs [14].

Because of the above-mentioned reasons, service composition is difficult. Researchers also have proposed some methods from different aspects.

Some researchers have given some service composition methods through recommendation. C. Xi et al. [15] use a novel collaborative filtering-based web service recommender system to help users to select the web service. The system takes account of the location information and QoS values and then clusters users and services. A recommendation is suggested based on the Cluster result. Q. Yu [16] proposes CloudRec to find out different community of users and services from large scale data. The service community is based on the inherent Cloud-related features, and it can help to estimate the QoS of unknown services. W. Denghui et al. [17] divide the QoS into two categories: global and local. Different methods are used to the two categories for a reliable web service composition recommendation approach. Global QoS dimensions are used to calculate the global QoS value of service composition. Then the credibility of service composition is computed with the distance of the execution result and the advertised QoS value. The lowest QoS value in all service units is selected in the local QoS dimension. The credibility of local QoS dimensions is got from the credible user experience. Web service evaluation result is the composition of the weight of user preference with the credible QoS information. The service with the highest score is chosen in the service selection.

Considering the preference of different users, researchers also give some new composition methods. To maximize (or at least ensure an acceptable) QoE, while ensuring fairness among users, H. Tobias et al. [4] propose a QoE fairness index F to solve this problem. S. Zhang et al. [18] try to help the user to find the top k composition services to satisfy the composition request, which has diverse preferences to different kinds of QoS. The preference-aware service dominance is used to denote the preference. A multi-index based algorithm is used to calculate the local service selection and top-k composite services are selected under a dominant number-aware service ranking in global optimization. According to

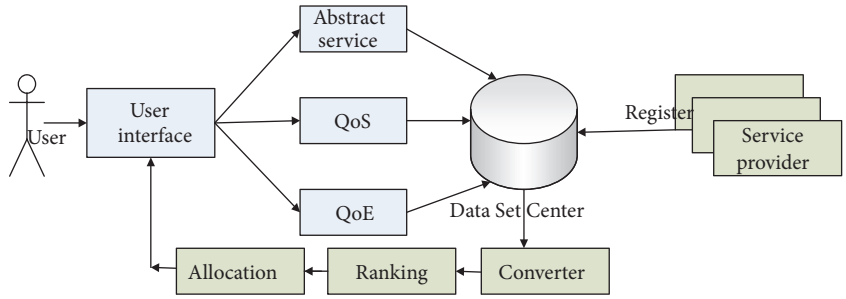


Figure 1: System overview of service composition.

the QoS preference of a user, H. Wang et al. [19] propose to integrate both the quantitative and qualitative preference to calculate the user similarity. They seek similar users by using user similarity service recommendation based on the idea of collaborative filtering.

Evolutionary algorithms are widely used in the web service composition. Y. Yu et al. [20] try to solve the service composition problem under distributed environment. They use a network coordinate system to estimate the time on the communication link between service providers and end users. And they propose a new GP-based (Genetic Programming) approach which considers multiple QoS constraints simultaneously. Z. Ding et al. [21] try to select and compose web services via a genetic algorithm (GA) and they combine those QoSs as a fitness function. They also give a transaction and QoS-aware selection approach. C. Cremene et al. [22] perform an analysis of several state-of-the-art multiobjective evolutionary algorithms. Those methods include Nondominated Sorting Genetic Algorithm II (NSGA-II), Strength Pareto Evolutionary Algorithm 2 (SPEA2), Multiobjective Multistate Genetic Algorithm (MOMS-GA), Differential Evolution (DE), multiobjective DE version called Generalized Differential Evolution (GDE3), and so on. Simulation results show that GDE3 algorithm yields the best performances on this problem.

Even the same value of QoS, various users have diverse QoEs. The reason is that different users have different targets. QoE also has been used in the service composition. Most of research focuses on the relation between the QoS and QoE. B. Upadhyaya et al. [6] provide an approach to extract a user's perception of the QoSs from user reviews on services (QoEs) and use such information to compose services. According to the QoEs of different users, they select a particular service from a pool of services and recommend the best service execution path in a composited service.

Some researchers also pay attention to the fact that some requirements of QoSs are flexible [8, 14]. In other words, if the value of the QoS attribute cannot satisfy the requirement, it just loses something, such as QoE and reputation. C. Lin et al. [8] proposed a QoS-based service selection (RQSS) algorithm to help users to find feasible web services according to the functionalities and the QoS of users' requirements. If no web service could exactly satisfy users' requirements, the RQSS recommends prospective service candidates to users by relaxing QoS constraints.

Other methods, such as ant colony [23] algorithm, liner approaches [24], and social network management [25], are also used in the service composition. Q. Wu et al. [23] model the problem of service composition problem as a constrained directed acyclic graph. They use the ant colony optimization algorithm to seek a near-to-optimal solution. S. Bharathan et al. [24] address the service composition in a multicloud environment. They propose an Integer Linear programming model where violations in global constraints are permitted but with an associated penalty. W. Chen et al. [25] use linked social service-specific principles based on linked data principles to publish services on the open web as linked social services. Based on complex network theories, they construct the global social service network following linked social service-specific principles. They used the global social service network to provide linked social services as a service.

Different from the prior work, we consider the environment when the requirements are a mixture of hard requirements and soft (relaxable) requirements. We try to discuss the service composition from the relation of QoS and QoEs, different from most of past work, which often composites services from the QoSs of the service. This also makes the task have diverse performance in QoE. S. Chakhar [26] just takes account of "hard" nonfunctional requirement in the scheduling after considering the function requirement. Different to [26], we try to consider the hard requirement and flexible (soft in [26]) requirement of functional requirement and nonfunctional requirement at the beginning. Most important of all, we consider the QoE of the service providers and service requesters together. We try to discuss the service composition from the QoEs of users, different from most of past work, which often composites services from the QoSs of the service.

3. SLA Service Composition Architecture

Our test data and the proposed method are implemented based on the framework shown in Figure 1. Users use "user interface" to submit the requirement of the abstract services, the requirement of QoSs, and the related QoE functions. The abstract services indicate which kind of services that users need. The requirement of the QoS illustrates the requirement of the value of QoS. The QoE function gives the relation between the QoS attribute and the QoE. In our paper, those requirements include hard requirement and soft requirement.

Service providers register service information to the system through UDDIe [27]. The information includes the identification of abstract services, the QoS of related services, the number of related services, and so on. The Data Set Center keeps and checks the information from the user and the service provider. The service provider registers service information (service name, QoSs) to the Data Set Center, and user provides the service requirement (service name, QoSs; QoEs) to Data Set Center through user interface. According to the QoS and the QoE function, Converter converts the QoS attribute value into QoE value. Different kinds of QoS requirement have different QoE function. Different to most of research, this paper ranks the service request and the service provider together which helps us to make decision in the Allocation (Section 4). Allocation schedules the web service providers to the web service requests (Section 5.3). The three modules (Converter, Ranking, and Allocation) work together to provide SLA (Service-Level Agreement) service composition for provider. A service-level agreement is a commitment between a service providers and users. SLA also supports the hard requirement and soft requirement. It makes various users may give diverse QoE value for the same QoS according to the SLA.

4. Web Service Ranking Based on Hard/Soft Requirement of Different Attributes

As discussed in Section 2, the web service has many attributes and even every attribute has many subattributes. The ranking problem is defined as multiple criteria decision making (MCDM). There are three fundamental approaches to solving MCDM problems: Analytic Hierarchy Process (AHP), Multiple Attribute Utility Theory (MAUT), and outranking. Most of methods are based on the three methods.

4.1. QoS Model. In the paper, we consider five QoSs and they are execution time, reliability, availability, reputation, and price [4, 8, 14, 17].

(i) Execution time: the execution time of the service request *sr* is the time from a service *s* has allocated to *sr* to the time that the *sr* has been finished.

(ii) Reliability: the reliability of the service *s* is the probability that a request is correctly responded as the request of the user in the expected time.

(iii) Availability: the availability of the service *s* is the probability that a service request exists or is ready to use within the expected time.

(iv) Reputation: reputation is used to reduce the impact of malicious attacks. The value of service reputation gets from customer's subjective scores and objective judgment to the credibility of QoS advertisement information (Bayesian Learning Theory) [4].

(v) Price: the cost is involved in requesting and using the service (Unit: USD/10 min).

More kinds of QoS, such as trust degree, operability, and delay, also can be added to our framework as the method introduced in the following.

There are M abstract services (formula (1)) and every abstract service has different numbers of candidate services. The abstract service s_m ($1 \le m \le M$) has ns_m candidate services (formula (2)). SC_m is the relative QoSs of different candidate services of the abstract service s_m. $sv(m, svid)$ denotes the $svid$th service candidate of the abstract service s_m (formula (3), $1 \le m \le M$). $QoS(m, svid)$ is the QoS set of the $svid$th service candidate of the abstract service s_m (formula (4), $1 \le svid \le ns_m$). $Et(m, svid)$, $Rel(m, svid)$, $Av(m, svid)$, $Rep(m, svid)$, and $Pr(m, svid)$ are the values of QoSs for the execution time, reliability, availability, reputation, and price, respectively.

$$S = \{s_1, s_2 \ldots, s_m, \ldots s_M\} \tag{1}$$

$$NS_m = \{ns_1, ns_2 \ldots, ns_m, \ldots ns_M\} \tag{2}$$

$$SC_m = \{sv(m,1), \ldots sv(m, svid), \ldots, sv(m, ns_m)\} \tag{3}$$

$$QoS(m, svid) = \{Et(m, svid), Rel(m, svid), Av(m, svid), Rep(m, svid), Pr(m, svid)\} \tag{4}$$

There are N service composition processes (requests) from the users (formula (5), $1 \le n \le N$). Formula (6) indicates which kind of abstract service the nth service composition process needs. If $sa(n, rid)$ equals 1 ($1 \le rid \le M$), it means that the nth service composition process needs the mth abstract service. At the same time, a vector of QoS requirements is listed in formula (7). $RQoS(n, rid)$ is the QoS requirement of nth service composition process to mth abstract service. $Ret(n, rid)$, $Rrel(n, rid)$, $Rav(n, rid)$, $Rrep(n, rid)$, and $Rpr(n, rid)$ are the requirement of the QoS of the execution time, reliability, availability, reputation, and price, respectively. Formula (8) computes the total number of candidate services to the abstract service s_m. M is the kinds of abstract services.

$$SP = \{sp_1, sp_2 \ldots, sp_n, \ldots sp_N\} \tag{5}$$

$$SA_n = \{sa(n,1), sa(n,2) \ldots, sa(n, rid), \ldots sa(n, M)\} \tag{6}$$

$$RQoS(n, rid) = \{Ret(n, rid), Rrel(n, rid), Rav(n, rid), Rrep(n, rid), Rpr(n, rid)\} \tag{7}$$

$$sum_m = \sum_{rid=1}^{M} sa(n, rid) \tag{8}$$

4.2. Different QoE to QoS. The value of QoE is a function to the related QoS of the candidate service. Figure 2 shows that some typical functions in the true system, X axis indicates the value of QoSs, and Y axis indicates the benefit for the user to the special QoS (QoE). Figures 2(a)–2(d) show the relations between the QoS and the QoE whose QoSs belong to benefit QoS (positive QoS) that higher value always brings better result. Figures 2(e)–2(h) show the QoS value that belongs to cost QoS (negative QoS) that higher value always brings negative effort to the user.

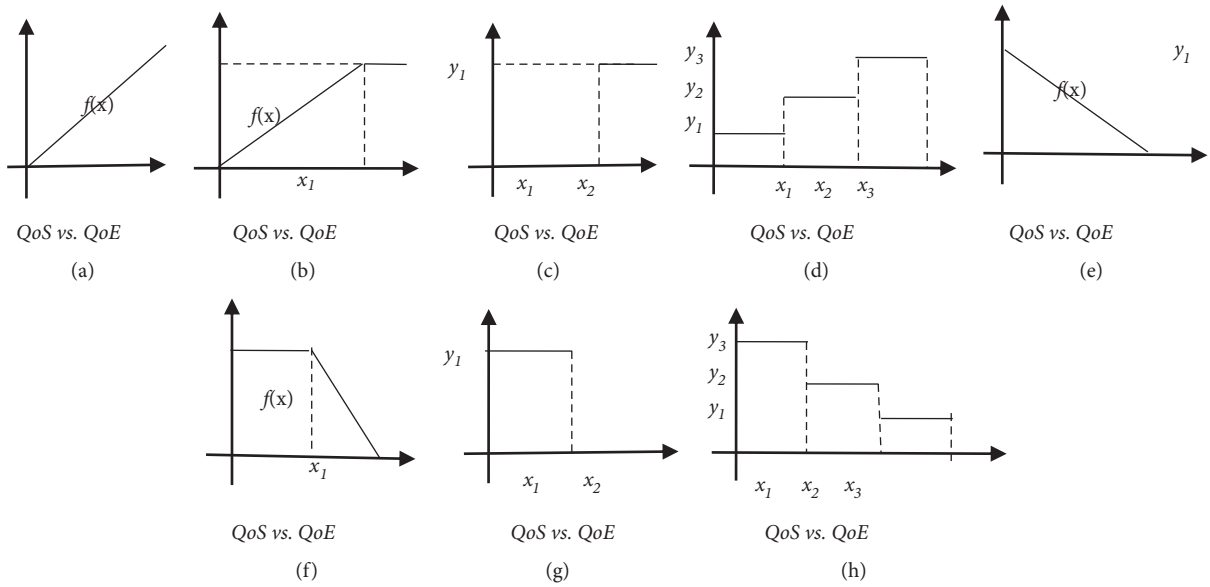


Figure 2: QoS versus QoE.

The QoE function $QoEv(m, svid, n, rid)$ of the QoS is

$$QoEv(m, svid, n, rid) = QoEf(QoS(m, svid), RQoS(n, rid)) \quad (9)$$

$QoS(m, svid)$ is the QoS set of the $svid$th service candidate of the abstract service s_m and $RQoS(n, rid)$ is the QoS requirement of nth service composition process to mth abstract service.

The value of QoE also can be categorized into four kinds: Boolean, numerical, unordered set, and range type. According to [28], same as the value of QoS, all of them can be transformed into numerical. In this paper, the hard requirement can be taken as the Figure 2(c) or Figure 2(g), and the soft requirement can be taken as Figure 2(b) or Figure 2(f).

5. QoS Normalization for QoSs of the Service Candidates and the Requirement of QoSs of the Service Requests

In this section, first of all, we normalize the QoS of service and the request to QoS of the service request, then we rank the service and service request; at last, a heuristic is proposed for service selection and composition.

5.1. QoS Normalization of the Service and the Service Request. The hypotheses in the paper are that, for the special abstract service s_m, the total number of the service candidates is ns_m (formula (2)), those service candidates are listed in the set SC_m (formula (3)), and the QoS of those service candidates is listed in the set $QoS(m, svid)$ (formula (4)); to the special service process sp_n, the request to different abstract services is listed in the set SA_n (formula (6)), and the requirement of the QoS of the abstract service is listed in the set $RQoS(n, rid)$ (formula (7)). Then the problem is how to assign services candidates to the service request under different kinds of requirement (hard and flexible) of the QoS. Past work mainly uses aggregation functions to schedule those services [4, 8, 9, 15]. A service composition request has many service requests to different abstract services. A service request refers to a detailed request to an abstract service. Before using the aggregation function, normalization of QoS is executed to make the values of different QoSs have the same scope (between 0 and 1).

Most of work for the normalization of the QoS is to the service, such as [8], not to the service request. In the paper, we consider two aspects simultaneously: QoS of service and the requirement of QoS of the service request. Such as execution time, we consider $Et(m, svid)$ $(1 \le svid \le ns_m)$ and $Ret(n, m)$ $(1 \le n \le N, 1 \le m \le M)$ as a new set ET (formula (10)). $Et(m, svid)$ and $Ret(n, m)$ are the requirement of execution time from the service requesters and the execution time from service providers.

$$ET = \{Et(m, svid) \mid 1 \le svid \le ns_m\} \cup \{Ret(n, m) \mid 1 \le m \le M\} \quad (10)$$

$Rel(n, m)$, $Rav(n, m)$, $Rrep(n, m)$, and $Rpr(n, m)$ mean the provided reliability, availability, reputation, and price from the services providers, respectively. $Rrel(n, m)$, $Rav(n, m)$, $Rrep(n, m)$, and $Rpr(n, m)$ mean the requirement of the QoS to reliability, availability, reputation, and price from service requesters, respectively. We make similar considerations for the other QoSs:

$$REL = \{Rel(m, svid) \mid 1 \le svid \le ns_m\} \cup \{Rrel(n, m) \mid 1 \le m \le M\}$$

$$\begin{aligned} AV &= \{Av(m, svid) \mid 1 \le svid \le ns_m\} \\ &\cup \{Rav(n, m) \mid 1 \le m \le M\} \\ REP &= \{Rep(m, svid) \mid 1 \le svid \le ns_m\} \\ &\cup \{Rrep(n, m) \mid 1 \le m \le M\} \\ PR &= \{Pr(m, svid) \mid 1 \le svid \le ns_m\} \\ &\cup \{Rpr(n, m) \mid 1 \le m \le M\} \end{aligned} \tag{11}$$

To make different values of QoSs of the service candidate and QoS requirements of the service request in the same range, Gaussian method [29, 30] is used to normalize them for feature extraction. It cannot only map those values to the interval [0, 1], but also better avoid the influence of abnormal values (such as high value or low value) compared with other normalization methods. The detail can be described as follows:

Suppose et_i^{Et} is an element of ET (formula (10)), we normalize it to no_i^{Et}:

$$no_i^{Et} = 0.5 + \frac{el_i^{Et} - \overline{ET}}{2 \times 3\sigma_{ET}} \tag{12}$$

where no_i^{Et} is normalized value of et_i^{Et}, $\overline{ET}$ is the average value of ET, and σ_{ET} is the standard deviation of ET. Prior work [30, 31] shows that 99% data can be mapped into [0, 1]. If the value is less than 0, we set it to 0; if the value is more than 1, we set it to 1.

Suppose no_i^{Rel}, no_i^{Av}, no_i^{Rep}, no_i^{Pr} are one element of REL, AV, REP, and PR, respectively. $\overline{REL}$, $\overline{AV}$, $\overline{REP}$, and $\overline{PR}$ are the average value of REL, AV, REP, and PR, respectively. σ_{REL}, σ_{AV}, σ_{REP}, and σ_{PR} are the standard deviation of REL, AV, REP, and PR, respectively. Similar to the method about execution time, we normalize the four elements as follows:

$$\begin{aligned} no_i^{Rel} &= 0.5 + \frac{el_i^{Rel} - \overline{REL}}{2 \times 3\sigma_{REL}} \\ no_i^{Av} &= 0.5 + \frac{el_i^{Av} - \overline{AV}}{2 \times 3\sigma_{AV}} \\ no_i^{Rep} &= 0.5 + \frac{el_i^{Rep} - \overline{REP}}{2 \times 3\sigma_{REP}} \\ no_i^{Pr} &= 0.5 + \frac{el_i^{Pr} - \overline{PR}}{2 \times 3\sigma_{PR}} \end{aligned} \tag{13}$$

If the value is not in the range [0, 1], we also use the same method to deal with the data to no_i^{ET}.

According to whether a larger value of QoS attribute means a better service, the QoS attributes can be categorized into two kinds: cost (negative) QoS or benefit (positive) QoS. Reliability, availability, and reputation belong to benefit QoS, execution time, and price belongs to cost QoS. For the cost QoSs, we transform them as follows so that make larger value of the related QoSs also means a better service:

$$\begin{aligned} not_i^{Et} &== 1 - no_i^{Et} = 1 - \left(0.5 + \frac{el_i^{Et} - \overline{ET}}{2 \times 3\sigma_{ET}}\right) \\ &= \frac{\overline{ET} - el_i^{Et}}{2 \times 3\sigma_{ET}} + 0.5 \\ not_i^{Pr} &= 1 - no_i^{Pr} = 1 - \left(0.5 + \frac{el_i^{Pr} - \overline{PR}}{2 \times 3\sigma_{PR}}\right) \\ &= \frac{\overline{PR} - el_i^{Pr}}{2 \times 3\sigma_{PR}} + 0.5 \end{aligned} \tag{14}$$

The above two formulas also have a scope of [0, 1].

5.2. Service Selection and Composition. For service composition, we give different weights to the five QoS attributes according to the condition of the whole system. Our model can extend to the services with more than five QoS attributes, but here we consider those important attributes only. The final rank of the service (or service request) S_m:

$$\begin{aligned} Sco_i^m = {} & W_{Et}^m \times not_i^{Et} + W_{Rel}^m \times no_i^{Rel} + W_{Av}^m \times no_i^{Av} \\ & + W_{Rep}^m \times no_i^{Rep} + W_{Pr}^m \times not_i^{Pr} \end{aligned} \tag{15}$$

i are the unique identifier of services (including service request and service provider). m is the unique identifier of abstract services. W_{Et}^m, $W_{Re;}^m$, W_{Av}^m, W_{Rep}^m, and W_{Pr}^m are the different weights of the five QoS attributes: execution time, reliability, availability, reputation, and price. Various researchers use diverse methods to set the weight. Those methods are classified into two kinds: weights decided by the user and those decided by the system. Our method belongs to the latter. Those parameters are listed as

$$\begin{aligned} tot = {} & \sum_{i=1}^{sum_m} Num_{i,m}^{Et} + \sum_{i=1}^{sum_m} Num_{i,m}^{Rel} + \sum_{i=1}^{sum_m} Num_{i,m}^{Av} \\ & + \sum_{i=1}^{sum_m} Num_{i,m}^{Rep} + \sum_{i=1}^{sum_m} Num_{i,m}^{Pr} \\ W_{Et}^m &= \frac{\sum_{n=1}^{sum_m} Num_{i,m}^{Et}}{tot} \\ W_{Rel}^m &= \frac{\sum_{i=1}^{sum_m} Num_{i,m}^{Rel}}{tot} \\ W_{Av}^m &= \frac{\sum_{i=1}^{sum_m} Num_{i,m}^{Av}}{tot} \\ W_{Rep}^m &= \frac{\sum_{i=1}^{sum_m} Num_{i,m}^{Rep}}{tot} \\ W_{Rp}^m &= \frac{\sum_{i=1}^{sum_m} Num_{i,m}^{Pr}}{tot} \end{aligned} \tag{16}$$

1. Calculate the number of all the service candidates (Sum_m);
2. $[M_{ID}, M_{value}] = sort(\{sum_m \mid 1 \le m \le M\})$;
   //sum_m is number of service candidates, sort sum_m in descending order;
   // M_{ID} is a set of the abstract service ID
   //M_{value} is a set of the number of the service requests to different abstract services.
3. For m=1:M
   3.1 $[mapc - r, unr]$ = schhard(abstract service s_m);
   //for the abstract service s_m, find a schedule method;
   //this step supposes that every requirement to the QoS is a hard requirement;
   //*mapc-r* is a map between the service candidate of the abstract service s_m and service request to the abstract service s_m;
   //*unr* is a service request set that cannot be satisfied;
   3.2 Delete the service process that includes the abstract service s_m and the request to the abstract service s_m cannot be satisfied;
   EndFor
4. For m=1:M
   4.1 $[mapc - r, unr]$ = schsoft(abstract service s_m)
   //this step considers the requirement to the QoS that is relaxable;
   //*mapc-r* is a map between the service candidate of the abstract service s_m and service request to the abstract service s_m;
   //*unr* is a service request set that cannot be satisfied;
   4.2 Delete the service process that includes the abstract service s_m and the request to the abstract service s_m that cannot be satisfied;
5. Endfor

Algorithm 1

where $\sum_{i=1}^{sum_m} Num_{i,m}^{Et}$ is the number of services which cannot satisfy the service request from the view of the execution time; sum_m (see formula (8)) is the total number of the request to the abstract service s_m.

$\sum_{i=1}^{sum_m} Num_{i,m}^{Rel}$, $\sum_{i=1}^{sum_m} Num_{i,m}^{Av}$, $\sum_{i=1}^{sum_m} Num_{i,m}^{Rep}$, and $\sum_{i=1}^{sum_m} Num_{i,m}^{Pr}$ are the numbers of services which cannot satisfy the service requests from the view of reliability, availability, reputation, and price, respectively.

5.3. Service Selection. There are two steps for the service selection: (1) the first step decides which abstract service will be allocated first; (2) the second step decides assign which service candidate to the service request of the selected abstract services.

Algorithm 1 gives the first step. First of all, we calculate the number of service requests to diverse abstract services (Line 1). Then, we sort the value in descending order. First of all, we schedule the abstract service which has the maximum value in the number of the service requests to different abstract service. The reason is if there are some service requests that cannot be satisfied, we can delete those service processes (Step 3.2, in Algorithm 1); those service processes maybe also include other service requests to other abstract services.

First of all, we suppose every requirement of the QoS attribute is a hard requirement; "*schhard*(abstract service s_m)" is used to find the service selection result (Steps 3.1 and 3.2, in Algorithm 1). Algorithm 2 gives the details. If there are service requests which cannot get the right services, soft requirement is considered and "*schsoft*(abstract service s_m)" is used to find the service selection result (Steps 4.1 and 4.2, in Algorithm 1). Algorithm 3 provides the details.

After we sort the SCO (line 3, in Algorithm 2) in descending order, we can consider the problem as two problems: selecting which service composition request and selecting which service candidate to the selected service request.

(1) The first problem decides which service request that we select first. "Largest First" is used to select the first service composition request. It means we always select the service request which has the largest value of the aggregation function. We select the "Largest First" because if the service request has a larger value in aggregation function, then it always has more difficult to find a service candidate that satisfies the requirement of the QoS attributes.

(2) The second problem decides which service candidate that we first try to check for the selected service request. "Local" means the service candidate whose position is near to the service request according to the value of aggregation function. In other words, there is no much difference in the value of aggregation function between those service candidates and the service requests. In the system, we give "Local" in this way: it has two directions, left or right. For the left, from the position of the service request, visiting left one by one, after it finds some service candidates and stops until it finds a service request or to the end of left again; for the right, from the position of the service request (Global view decides the service request), visiting right one by one, after it finds some service candidates or to the end of the right and stops until it finds a service request again. For the local view, we can check the best service (Local Best First Check) candidate. The service candidate who has a higher value of aggregation

```
//GLLB
1. Calculate the weights of different QoSs according to formulas in Sections 5.1 and 5.2;
2. Calculate the aggregation value for the service candidate and service request of the abstract
   service s_m;
3. SCO = {Sco_i^m | 1 ≤ i ≤ (ns_m + sum_m)};
4. [tp, ID, Val = sort(SCO)];
      // sort SCO in descending order;
      //tp can be a service candidate (="candidate") or a service request (="request");
      //ID records the ID for the service candidate or the service request;
      //Val records the value of aggregation function for the service candidate or the service
request;
5. Select the first service composition request sr
6. Get the left position lp and the right position rp;
7. minmore=+∞, selectr=-1;
8. For pos = lp : rp
        If the service candidate Sco_pos^m can satisfy sr for all the QoS
            minmoretemp = mvalue(Sco_pos^m , sr);
          If minmoretemp< minmore
            minmore=minmoretemp;
            selectr=Sco_pos^m;
          Endif
        EndIf
   Endfor
9. If selectr=-1
        Assign selectr to sr;
   EndIf
   //minmore records the total value of the QoS of the web service that which is more than the
requirement of the service request under the condition that all of requirements of the service
composition request have been satisfied.
10. pos = rp + 1;
        //search from the position of the end of the right position;
11. While (pos < (ns_m + sum_m))
            11.1 If Sco_pos^m can satisfy the requirement of QoS when we suppose every
            requirement is hard
                    selectr=Sco_pos^m;
                    Assign selectr to sr;
                    Break;
                EndIf
   EndWhile
```

Algorithm 2: Schhard (abstract service s_m).

value does not always mean it can satisfy the service request with lower aggregation value. Some service requests may have the hard requirement of the QoS. Even some service candidates which have a lower value of aggregation function also can satisfy the service request which has a higher value of aggregation function by relaxing some requirements of soft QoSs. If a service request cannot get a service candidate in the left and the right, it will search all service candidates.

In summary, our method is GLLB (global largest number of service request first, local "best fit" service candidate first). For the service composition request, the one which has the largest aggregation function value will be selected first; the service candidate, the one which is the "best fit", will be selected. "Best fit" has two aspects of meaning.

When we suppose every requirement of QoS attribute is hard, we select the service candidate that would have the lowest lost value (See formulas (18)~(22)). We suppose there are a web service candidate *wsc* and a web service request *wsr*, and QoSs of wsc and QoS requirement of *wsr* have been normalized. $hard_{wsr}$ indicates the requirement of the related QoS that is relaxable. If ETh_{wsr} equals 0, it is a hard requirement; otherwise (being equal to 1), it is a flexible requirement, similar to others QoS attributes. If there are many service candidates that can satisfy the requirement when we suppose they are hard requirements, we will select the service candidate which has the smallest value in *mvalue*(formula (20)), which has the smallest average lost value in the QoS. *moren* is the number of QoSs which is better than the requirement. Formula (21) calculates the lost value of execution time, same to all cost QoS attributes; formula (22) calculates the lost value of reliability, same to others benefit QoS attributes. If we cannot get a service candidate, we will go to the next step.

$$wsc = \{ET_{wsc}, Rel_{wsc}, Av_{wsc}, Rep_{wsc}, Pr_{wsc}\} \quad (17)$$

```
// GLLB
1. Calculate the weight of different QoS according to formulas in Sections 5.1 and 5.2;
2. Calculate the aggregation value for the service candidate and service request of the abstract
   service s_m;
3. SCO = {Sco_i^m | 1 ≤ (ns_m + sum_m)};
4. [tp, ID, Val] = sort(SCO);
      //sort SCO in descending order;
      //tp can be a service candidate (="candidate") or a service request ("request");
      //ID indicates the one for the service candidate or the service request;
      //Val indicates the value of aggregation function for the service candidate or the service
request;
5. Select service composition request sr that cannot find a service candidate in Algorithm 2
   (Algorithm 2)
6. Get the left position lp and the right position rp;
7. minlost=+∞, selectr=-1;
8. For pos = lp : rp
        If service candidate Sco_pos^m can satisfy sr for all the QoS
           minlostemp=slost(Sco_pos^m , sr);
           If minlosttemp< minlost
              minlost=minlosttemp;
              selectr=Sco_pos^m;
           Endif
        EndIf
   Endfor
9. If selectr=-1
        Assign selectr to sr;
   EndIf
      //minmore records the total value of the QoS of the web service that is more than the
requirement of the web service request under the condition that all of the hard requirements of
the service request have been satisfied.
10. pos = rp + 1;
      //search from the position of the end of the right position;
11. While (pos < ns_m + sum_m)
      11.1 If Sco_pos^m can satisfy the hard requirement of QoS of sr
              selectr=Sco_pos^m;
              Assign selectr to sr;
              Break;
           EndIf
   EndWhile
```

Algorithm 3: Schsoft (abstract service s_m).

$$wsr = \{ET_{wsr}, Rel_{wsr}, Av_{wsr}, Rep_{wsr}, Pr_{wsr}\} \tag{18}$$

$$hard_{wsr} = \{ETh_{wsr}, Relh_{wsr}, Avh_{wsr}, Reph_{wsr}, Prh_{wsr}\} \tag{19}$$

$$mvalue\left(QoS_{wsc}, QoS_{wsr}\right) = \frac{\sum more\left(QoS_{wsc}, QoS_{wsr}\right)}{moren} \tag{20}$$

$$more\left(ET_{wsc}, ET_{wsr}\right) = \begin{cases} +\infty, & if\ ET_{wsc} \le ET_{wsr}; \\ ET_{wsc} - ET_{wsr}, & if\ ET_{wsc} > ET_{wsr}; \end{cases} \tag{21}$$

$$more\left(Rel_{wsc}, Rel_{wsr}\right) = \begin{cases} +\infty, & if\ Rel_{wsc} \ge Rel_{wsr}; \\ Rel_{wsr} - Rel_{wsc}, & if\ Rel_{wsc} < Rel_{wsr}; \end{cases} \tag{22}$$

For the service requests which have not been assigned web services when we suppose every requirement of QoS attributes is hard, we will consider the soft requirement of QoS attributes. We add a new parameter to calculate the loss of the soft QoS attribute *(slost)* when a web service candidate cannot satisfy the requirement of the relaxable QoS attribute. Formula (23) is used to calculate the loss for the cost QoS attribute (execution time, similar to price) and formula (24) is used to calculate the loss for the benefit QoS attribute (reliability, similar to availability, and reputation). In fact, we

take the lost value as the QoE of the scheduling. Formula (25) gives the totally lost value in the soft QoS attribute. Formula (26) gives the method of calculating of average loss of soft requirements. $sofutnum$ is the number of the resources whose QoS attributes do not satisfy the requirement. The related formulas are as follows:

$$lost\left(ET_{wsc}, ET_{wsr}\right) = \begin{cases} 0, & if\ ET_{wcs} \le ET_{wsr}; \\ ET_{wsc} - ET_{wsr}, & if\ ET_{wsc} > ET_{wsr}; \end{cases} \tag{23}$$

$$lost\left(Rel_{wsc}, Rel_{wsr}\right) = \begin{cases} 0, & if\ Rel_{wsc} \ge Rel_{wsr}; \\ Rel_{wsr} - Rel_{wsc}, & if\ Rel_{wsc} < Rel_{wsr}; \end{cases} \tag{24}$$

$$\begin{aligned} lostvalue\left(wsc, wsr\right) = {} & lost\left(ET_{wsc}, ET_{wsr}\right) + lost\left(Rel_{wsc}, Rel_{wsr}\right) + lost\left(Av_{wsc}, Av_{wsr}\right) + lost\left(Rep_{wsc}, Rep_{wsr}\right) \\ & + lost\left(Pr_{wsc}, Pr_{wsr}\right) \end{aligned} \tag{25}$$

$$slost\left(wsc, wsr\right) = \begin{cases} +\infty, & if\ there\ are\ hard\ requirements\ cannot\ be\ satisified; \\ \dfrac{lostvalue}{sofutnum}, & if\ there\ are\ soft\ requirement\ cannot\ be\ satisfied; \end{cases} \tag{26}$$

The details of our scheduling algorithms are listed in Algorithms 2 and 3. Step 3.1 (in Algorithm 1) is the second step which is in charge of scheduling the abstract service s_m. "schhard(abstract service s_m)" (Algorithm 2) is the service composition method when we suppose every requirement of the QoS attribute is a hard requirement. First of all, we calculate the weight of different QoSs according to formulas in Section 5.1 (Step 1, in Algorithm 2), and calculate the aggregation value of the service candidate and the service request use the formulas in Section 5.2 (Step 2, in Algorithm 2). Then, we list all the values in a set *SCO*, and we sort the *SCO* in descending order. Three values are returned and they are sets *tp*, *ID*, and *Val*. "*tp*" has two possible values, if it equals "*candidate*"; it is a service candidate of the abstract service s_m; at this time, the value in *ID* is the service candidate and *Val* is the related value of aggregation function; if it equals "*request*", it is a service request to the abstract service s_m. At this time, the value of *ID* is the service request and *Val* is the related value of aggregation function. First, we will search the best fit service candidate between the left position *lp* and the right position *rp* (Step 6 -9, in Algorithm 2). *minmore* records the minimum value in the function of *mvalue* and *selectr* records the selected service candidates. If we can find some service candidates that can satisfy all the requirement when we suppose every requirement of the QoS attribute is a hard requirement, we will select the service candidate which has minimum value in the function of *mvalue* (Step 8, in Algorithm 2; formula (20)). If we cannot get a service candidate between the left position and the right position, we will search from the end of the right position until we find a service candidate which can satisfy all the requirements of the QoS under the assumption that every requirements of the QoS attribute which belongs to the hard requirement (Step 11, in Algorithm 2) have been satisfied.

Algorithm 3 is the service selection method for the soft requirement of QoS attributes under the method of GLLB. We will first search the best fit service candidate between the left position *lp* and the right position *rp* (Steps 6-9, in Algorithm 3). *minlost* records the minimum value in the function of *slost* (Formula (26)) and *selectr* records the selected service candidate. If we can find some service candidates that can satisfy all the requirement when we suppose every requirement of the QoS attribute is a hard requirement, we will select the service candidate which has minimum value in the function of *slost* (Step 8, Algorithm 3; formula (26)). If we cannot get a service candidate between the left position and the right position which can satisfy the hard requirement of the QoS attribute, we will search from the end of the right position until we find a service candidate which can satisfy all the requirement of the QoS, under the supposition that we can give up some soft requirements (Step 11, Algorithm 3).

5.4. Complexity Analysis of GLLB. We analyze our method for one special abstract service s_m from three steps:

(1) Calculating the weight of different QoSs (formulas in Section 5). The number of the candidate services is sum_m, the number of service requests is num_m, and we only pay attention to five QoS attributes. So, the complexity is $O(5 * num_m * sum_m)$.

(2) Computing the normalized quality values of all service candidates and service requests. The complexity of calculating average value is $O(num_m * sum_m)$, and the complexity of calculating standard deviation is also $O(num_m * sum_m)$. We take account of five QoS attributes, so the complexity is $O(5 * num_m * sum_m)$.

(3) The core of GLLB (Algorithms 2 and 3). Under the worst case, every service request will check every service candidate, so the complexity is $O(num_m * sum_m)$.

Table 1: Values of five QoSs.

QOS	QoS Range	QoS Constraint
Execution time	[1,100]	[1+99∗*CF*,100]
Reliability	[0.95,0.9999]	[0.95,0.9999-0.0499∗*CF*]
Availability	[0.95,0.9999]	[0.95, 0.9999-0.0499∗*CF*]
Reputation	[1,10]	[1, 10-9∗*CF*]
Price	[1,100]	[1+99∗ *CF*,100]

And thus, in generally, the complexity of our method is

$$\begin{aligned} &O(5 * num_m * sum_m) + O(5 * num_m * sum_m) \\ &\quad + O(num_m * sum_m) = O(num_m * sum_m) \end{aligned} \tag{27}$$

For all the abstract services, suppose the total number of the abstract services is M; the complexity is $O(M * num_m * sum_m)$.

6. Experiment Results and Discussions

In the simulation, we will compare our method to RQSS [8], MDSC (Multidimension Service Composition) [28], and RASC (Rank Aggregation Service Composition) [32]. RQSS [8] helps user discover feasible web services based on functionalities and QoS requirements. RQSS recommends prospective service candidates to users by relaxing QoS constraints if no available web service could exactly fulfill users' requirements. MDSC [28] first normalizes those QoS parameters and then uses the improved Euclidean Distance to select the web service which meets the requirement (in the simulation, we suppose every part has the same weight). RASC [32] uses Rank Aggregation methods to solve the web service composition.

Although there are many kinds of structures of the service composition, such as the cycle, sequence structure, parallel structure, loop structure, and case structure, we just pay attention to the sequence structure. All other structures can be transformed into sequence structures by some methods [28, 29].

6.1. Simulation Environment. We utilize the Matlab language to implement these algorithms and ran them on an Intel Pentium (R) D 3.4 Ghz, 4 GB RAM desktop PC with 100 MB/s Ethernet card, Window 8. In the simulation, we consider five QoS attributes which have been discussed before: execution time, reliability, availability, reputation, and price. The value of each QoS attribute is randomly generated with a uniform distribution in a range, as shown in Algorithm 3. These ranges are also used in some researches [28]. In the simulation, we assume that the execution times of all services are less than 100 *ms*. And most of services are robust enough. Therefore, the reliability and availability of service are ranging from 0.95 to 0.99999. Those settings are reasonable for web services [32]. In addition, the reputation has range of [1, 10] and the price has range of [1, 100]. These assumptions are also used in [8] and they are reasonable in various applications and different environments.

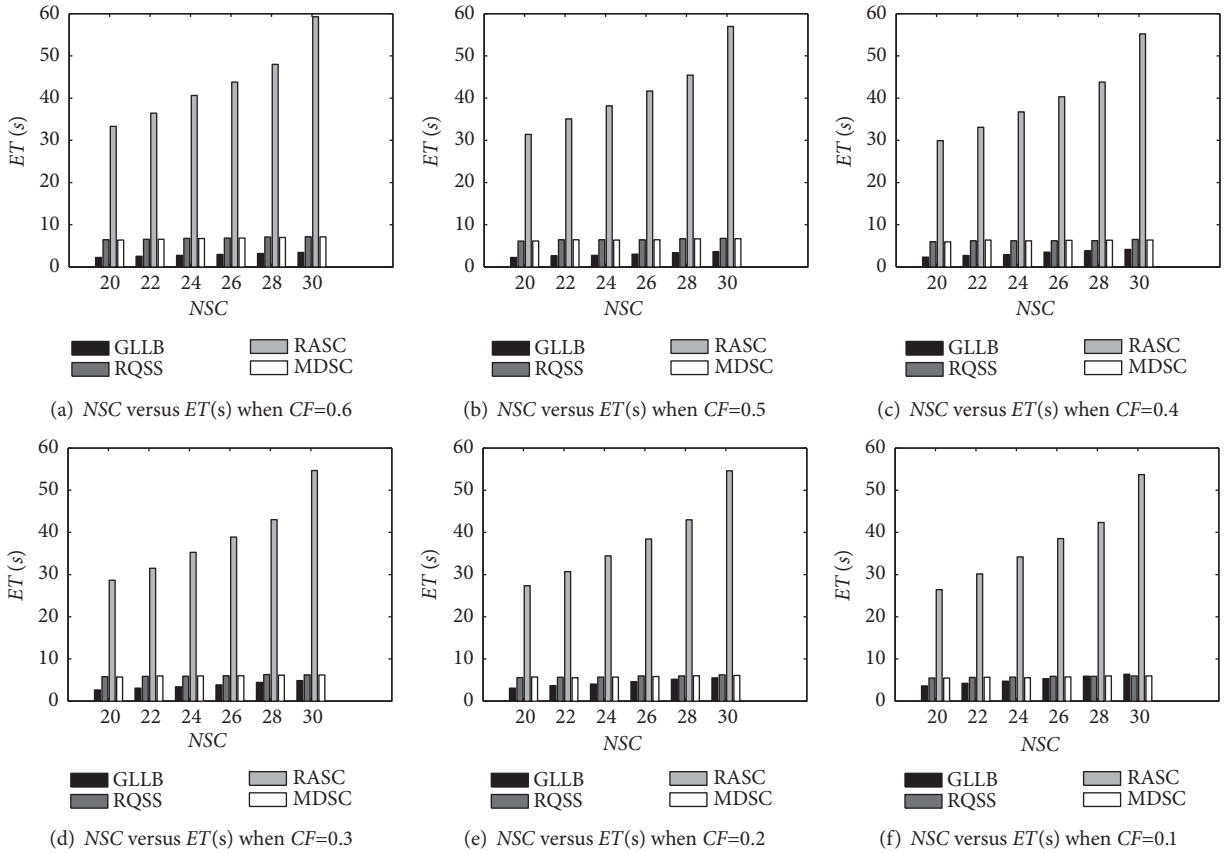
The QoS constraint is generated as the last column in Table 1. The method also was used in [28], but there is some difference. *CF* is the constraint factor and it has a range of [0, 1]. The values of relative QoS attributes are random in the range of the last column in Table 1 and they follow the uniform distribution. It can be used to represent the strength of a QoS constraint. If a requirement of the QoS is a soft one: a lower value of *CF* always means a lower of the QoS constraint (the QoE is same to the Figure 2(b) or Figure 2(f)). When the QoS is in the scope of the last column of Table 1, the QoE of relative QoS is 1. If a requirement of the QoS is a hard one, the QoE is the same to Figure 2(c) or Figure 2(g). If the QoS is not in the scope of the last column of Table 1, the QoE would be 0; otherwise, it is 1.

We will give comparisons from the execution time (*ET*), the number of the unfinished web service requests (*NUR*), the total value of failure to QoS attributes (*FV*), and the average failure to the QoS attributes (*AFV*). There are 500 abstract services and 2000 service processes to those abstract services. The possibility of a service process including an abstract service is 1%. We will evaluate the value of *CF* changed from 0.6 to 0.1 with a step of -0.1. The number of service candidates to every abstract service is changed from 20 to 30 with a step of 2. Every QoS has 75% possibility that it belongs to the hard requirement. The evaluated values in the following section are the average values of 50 times executions.

6.2. Execution Time. Figure 3 is the execution times of different methods under various numbers of service candidates and diverse values of *CF*. The values of X axes are NSC (number of service candidates) and the values of Y axis are execution times. All methods have the same trends: *ET* increases with the increasing of *CF* and the number of service candidates. The reason is that, with the increasing of *CF*, the service request needs to check more service candidates, and with the increasing of the number of the service candidates, the service request has more chances to select.

In general, RASC always has the largest value in *ET* and its average value under all the conditions is 39.8603 (s); GLLB always has the lowest value in *ET* and its average value is 3.6867 (s). GLLB only has 9.25% execution time to RASC. To RQSS and MDSC, the GLLB average value is reduced by 59.59% and 59.70%, respectively.

RASC has the largest value in *ET*, the reason is the calculations of the Kendall Tau Distance between the service request and the service candidates needs much time [32]. The execution time is a polynomial function to the number of service candidates. RQSS and MDSC do not lead to any difference in the execution time, and the execution times



(a) *NSC* versus *ET*(s) when *CF*=0.6 (b) *NSC* versus *ET*(s) when *CF*=0.5 (c) *NSC* versus *ET*(s) when *CF*=0.4

(d) *NSC* versus *ET*(s) when *CF*=0.3 (e) *NSC* versus *ET*(s) when *CF*=0.2 (f) *NSC* versus *ET*(s) when *CF*=0.1

Figure 3: The execution time of different methods.

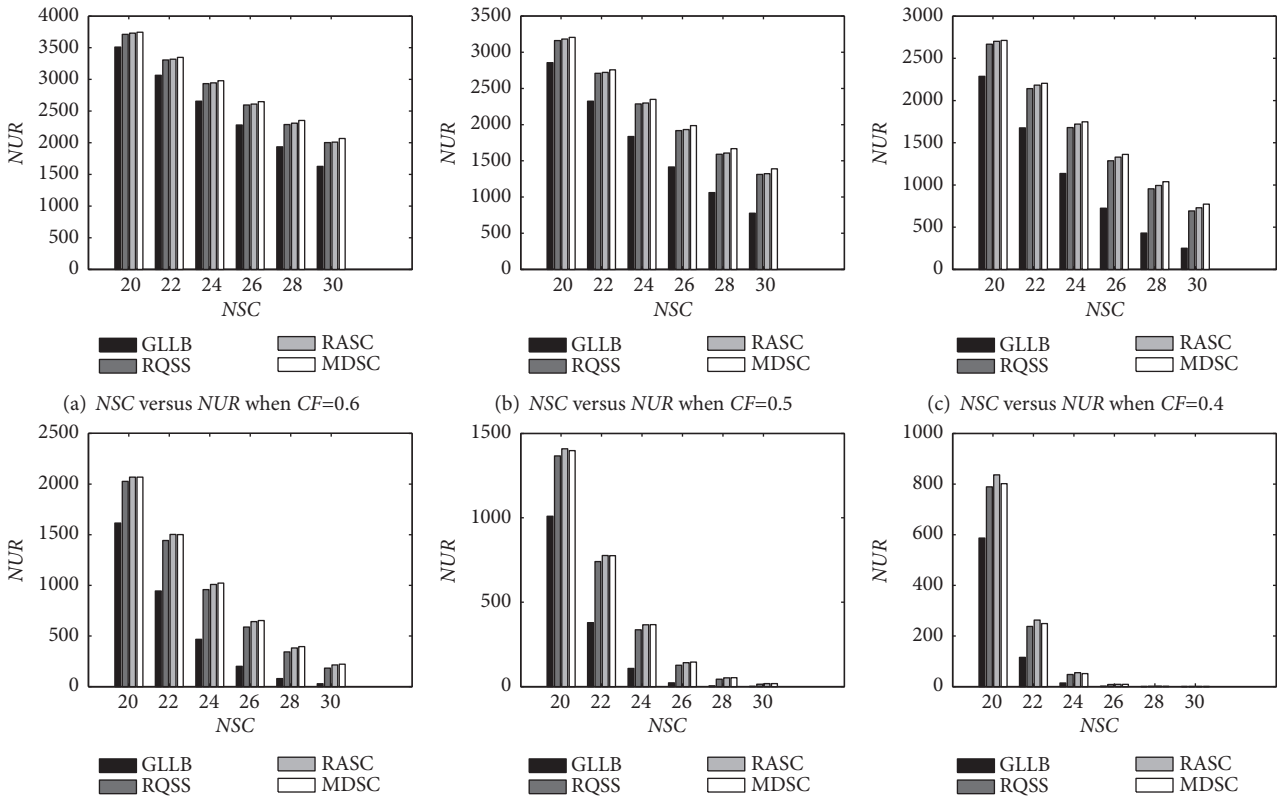
of them are a linear function to the number of the service candidates. The ranking of GLLB reduces the time to search all the service candidates. Most of time, GLLB only needs searching between the left and the right of the service request. By the way, in the simulation, we may spend some time to get the normalize value (Section 5.1), but we can get in before scheduling, so the execution time of GLLB does not include the time to get the relative normalize time. A tradeoff way is to get those data (average value; standard deviation) from the past record.

6.3. Comparison of the Number of Unfinished Web Service Requests in Finding a Feasible Composition. Figure 4 shows the number of unfinished service requests (*NUR*) of the four methods under different numbers of service candidates and various values of *CF*. The values of X axes are the number of service candidates and the values of Y axes are the values of *NUR*. The four methods involved in the comparison have the same patterns: *NUR* drops with the reducing of the number of service candidates and the value of *CF*. The reasons are as follows: with the dropping of the number of service candidates, there are fewer service candidates for the service request; at the same time, with the dropping of *CF*, the stronger requirements of QoSs are made, and the more service requests cannot get the required services. GLLB always has the lowest value in *NUR* and it has an average value of 1039. The other three methods have the same performance in *NUR* basically. The average *NUR* of RQSS, RASC, and MDSC is 1346, 1372, and 1390, respectively. To RQSS, RASC, and MDSC, the GLLB average id reduced by 22.81%, 24.27%, and 25.25%, respectively.

GLLB has the lowest value in *NUR* because of the policy: when we suppose, every requirement is a hard requirement, we try to find the service candidate which has the nearest value to all the QoS, the policy makes the best fit service candidate always be selected first, and it makes the left service requests have more opportunities to get the required service candidates.

6.4. Comparison of Average Violated Quality Value (AVQV) to the QoS Requirement. Figure 5 shows the failure value to the QoS requirement (*FV*) of the four methods under different numbers of service candidates and different values of *CF*. The values of X axes are the number of service candidates and the value of Y axis are the value of *FV*. The four methods behave same patterns: they all drop with the reducing of the number of service candidates and the value of *CF*.

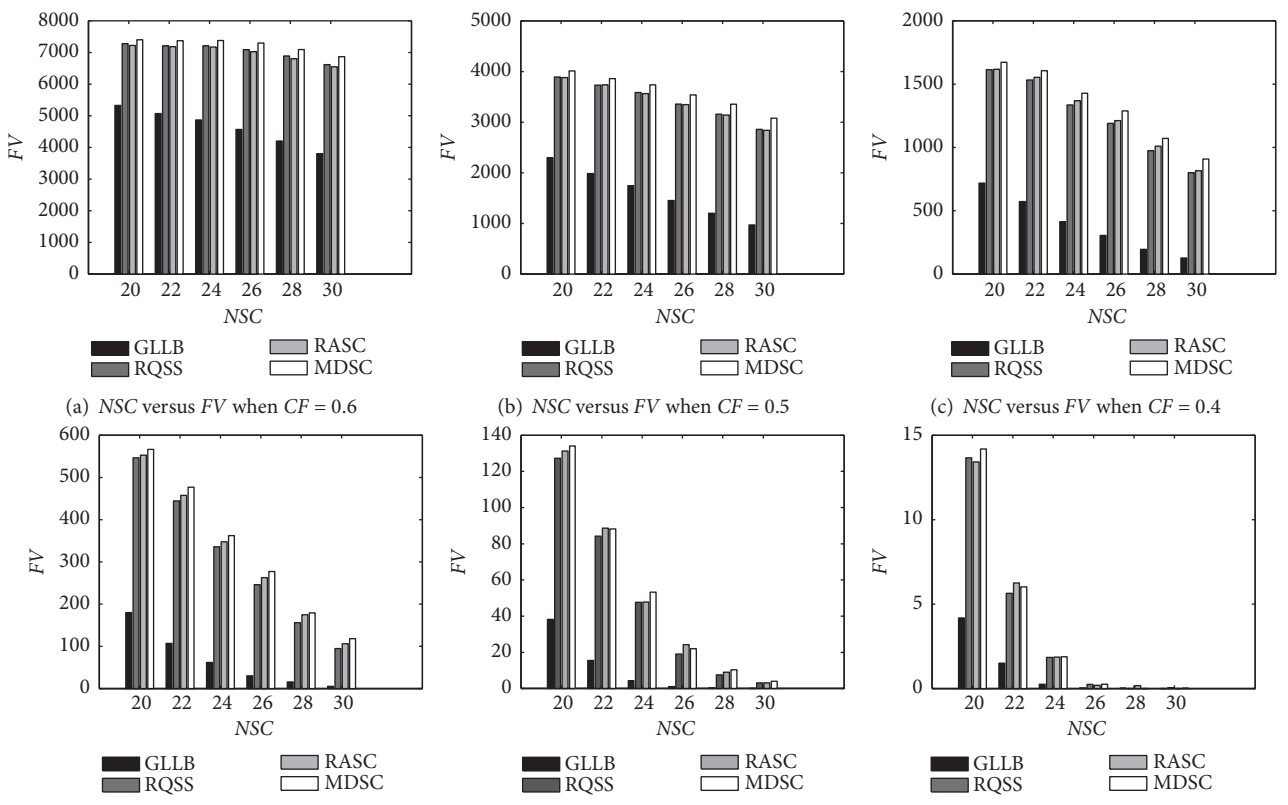
GLLB always has the lowest value in *FV* and it has an average value of 1190. The other three methods do not show much different performance in *FV*. The average *FV* of RQSS, RASC, and MDSC is 2012, 2007, and 2091, respectively. To RQSS, RASC, and MDSC, the average value of FV in the case of using GLLB is enhanced by 79.80%, 79.36%, and 86.86%, respectively.



(a) *NSC* versus *NUR* when *CF*=0.6 (b) *NSC* versus *NUR* when *CF*=0.5 (c) *NSC* versus *NUR* when *CF*=0.4

(d) *NSC* versus *NUR* when *CF*=0.3 (e) *NSC* versus *NUR* when *CF*=0.2 (f) *NSC* versus *NUR* when *CF*=0.1

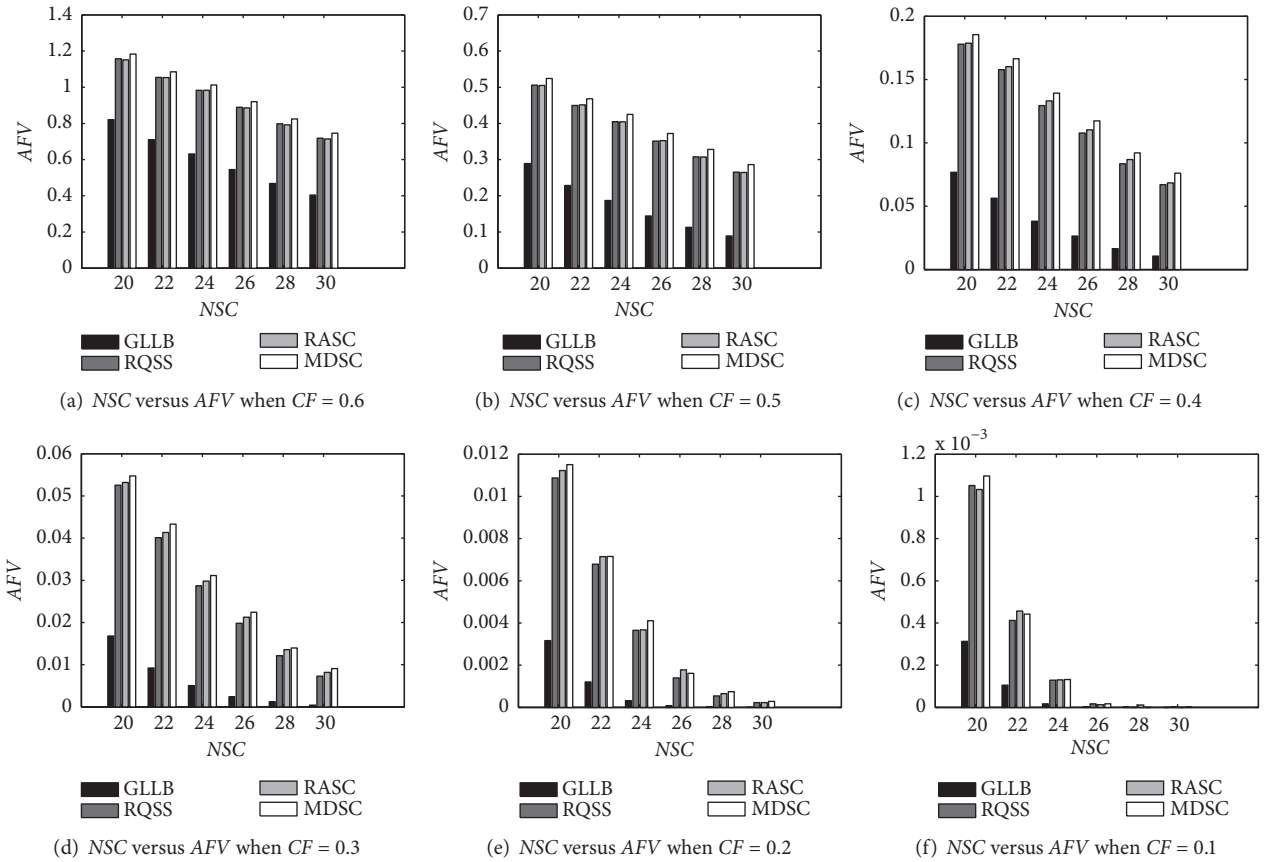
Figure 4: The number of unfinished web service requests.



(a) *NSC* versus *FV* when *CF* = 0.6 (b) *NSC* versus *FV* when *CF* = 0.5 (c) *NSC* versus *FV* when *CF* = 0.4

(d) *NSC* versus *FV* when *CF* = 0.3 (e) *NSC* versus *FV* when *CF* = 0.2 (f) *NSC* versus *FV* when *CF* = 0.1

Figure 5: The total Failure value to the soft QoSs of different methods.



(a) *NSC* versus *AFV* when *CF* = 0.6
(b) *NSC* versus *AFV* when *CF* = 0.5
(c) *NSC* versus *AFV* when *CF* = 0.4
(d) *NSC* versus *AFV* when *CF* = 0.3
(e) *NSC* versus *AFV* when *CF* = 0.2
(f) *NSC* versus *AFV* when *CF* = 0.1

Figure 6: The Average failure value to the soft QoSs of different methods.

GLLB has the lowest value in *FV* because: when we suppose all the requirement of the QoSs are hard, we try to find the service candidates which are the nearest to the service candidates; this makes other service requests have more chances to get the right service candidate which can satisfy all the requirement; no matter they belong to the hard requirement or the soft requirement; if GLLB finds that a service process cannot get the right service candidate in the first step, we always try to find the service candidate when the scheduling has the lowest value in *FV*.

Figure 6 shows the average failure value to the QoS requirement (*AFV*) of the four methods under different numbers of service candidates and different value of *CF*. The value of X axis is the number of service candidates and the value of Y axis is the value of *AFV*. The four methods show the same patterns: they all have a dropping tendency when the number of service candidates and the value of *CF* are reducing.

GLLB always has the lowest value in *AFV* and it has an average value of 0.1360. The other three methods do not show much different performance in *AFV*. The average *AFV* of RQSS, RASC, and MDSC is 0.2443, 0.2444, and 0.2543, respectively. *AFV* of GLLB is only about 55.67%, 55.65%, and 53.48% to the *AFV* of RQSS, RASC, and MDSC.

GLLB has the lowest value in *AFV* because it has the lowest value in *FV* (Figure 5) and it also has the lowest value in *NUR* (Figure 4).

7. Conclusion

In the paper, we analyze the service composition problem from the view of QoS and QoE. Then, we rank the service composition requests and the services in order. Based on the rank of them, a service selection and composition method is proposed to meet the environment when there are soft and hard requirements of the QoS attributes of web services. First of all, we take all the requirements of QoS attributes as the hard requirement, if we cannot get the right service candidate; we try to find service candidate by relaxing some QoSs under the permission of the service request.

In the Cloud, when there are many Clouds and every Cloud has various kinds of abstract services and different numbers of related abstract services, the service composition problem becomes more difficult. Different QoSs have diverse "overload" values [33]. For the multiple Clouds environment [34], more services and more choices bring more challenge to service composition. We not only want to consider the requirement of every web service, but also want to reduce the number of the related Clouds that involved in the service composition. The problem of service composition under multiple Clouds [34] is the future work of us. Because we need to normalize the QoS, we may optimize our method to ensure them work well just from a small local scope and then work for overall situation. In this paper, we just evaluate our work on a simulation environment; as a future work, we also hope

we can evaluate our work on a real web service composition case.

Data Availability

Below is the weblink where other researchers can download the data: https://pan.baidu.com/s/12NjYa8q5wrV4AmcyVc7XYA.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work was partly supported by the National Natural Science Foundation of China (NSF) under Grant no. 41475089 and major projects of the National Social Science Fund (16ZDA054), the Natural Science Foundation of Zhejiang Province (LQ18F020005), and Science and Technology of Wenzhou (S20170008).

References

[1] Y. Hao, L. Wang, and M. Zheng, "An adaptive algorithm for scheduling parallel jobs in meteorological Cloud," *Knowledge-Based Systems*, vol. 98, pp. 226–240, 2016.

[2] S. K. Garg, S. Versteeg, and R. Buyya, "A framework for ranking of cloud computing services," *Future Generation Computer Systems*, vol. 29, no. 4, pp. 1012–1023, 2013.

[3] A. Ramírez, J. R. Romero, A. Ruiz-Cortés et al., "Evolutionary composition of QoS-aware web services: A many-objective perspective," *Expert Systems with Applications*, vol. 72, pp. 357–370, 2017.

[4] V. Gabrel, M. Manouvrier, K. Moreau, and C. Murat, "QoS-aware automatic syntactic service composition problem: Complexity and resolution," *Future Generation Computer Systems*, vol. 80, pp. 311–321, 2018.

[5] H. Kurdi, A. Al-Anazi, C. Campbell, and A. A. Faries, "A combinatorial optimization algorithm for multiple Cloud service composition," *Computers & Electrical Engineering*, vol. 42, pp. 107–113, 2015.

[6] B. Upadhyaya, Y. Zou, J. Ng, T. Ng, and D. Lau, "Towards quality of experience driven service composition," in *Proceedings of the 2014 IEEE World Congress on Services (SERVICES)*, pp. 18–20, Anchorage, AK, USA, June 2014.

[7] A. Jula, E. Sundararajan, and Z. Othman, "Cloud computing service composition: a systematic literature review," *Expert Systems with Applications*, vol. 41, no. 8, pp. 3809–3824, 2014.

[8] C.-F. Lin, R.-K. Sheu, Y.-S. Chang, and S.-M. Yuan, "A flexibleservice selection algorithm for QoS-based web service composition," *Information and Software Technology*, vol. 53, no. 12, pp. 1370–1381, 2011.

[9] N. Anithadevi and M. Sundarambal, "A design of intelligent QoS aware web service recommendation system," *Cluster Computing*, vol. 1, pp. 1–10, 2018.

[10] Y. Feng, L. D. Ngan, and R. Kanagasabai, "Dynamic service composition with service-dependent QoS attributes," in *Proceedings of the 2013 IEEE 20th International Conference on Web Services, ICWS 2013*, pp. 10–17, USA, July 2013.

[11] R. Iordache and F. Moldoveanu, "A web service composition approach based on QoS preferences," in *Proceedings of the 6th IEEE International Conference on Service-Oriented Computing and Applications, SOCA 2013*, pp. 220–224, USA, December 2013.

[12] G. Liu, Y. Wang, and M. A. Orgun, "Finding K optimal social trust paths for the selection of trustworthy service providers in complex social networks," in *Proceedings of the 2011 IEEE 9th International Conference on Web Services, ICWS 2011*, pp. 41–48, USA, July 2011.

[13] L. Li, Y. Wang, and E. P. Lim, "Trust-oriented composite service selection with QoS constraints," *Journal of Universal Computer Science*, vol. 16, no. 13, 2010.

[14] T. Hofeld, L. Skorin-Kapov, P. E. Heegaard et al., "A new QoE fairness index for QoE management," *Quality & User Experience*, vol. 3, no. 1, p. 4, 2018.

[15] X. Chen, Z. Zheng, Q. Yu, and M. R. Lyu, "Web service recommendation via exploiting location and QoS information," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1913–1924, 2014.

[16] Q. Yu, "CloudRec: a framework for personalized service Recommendation in the Cloud," *Knowledge and Information Systems*, vol. 43, no. 2, pp. 417–443, 2015.

[17] W. Denghui, H. Hao, and X. Changsheng, "A novel web service composition recommendation approach based on reliable QoS," in *Proceedings of the IEEE 8th International Conference on Networking, Architecture and Storage (NAS '13)*, pp. 321–325, IEEE, Xi'an, China, July 2013.

[18] S. Zhang, W. Dou, and J. Chen, "Selecting top-k composite web services using preference-aware dominance relationship," in *Proceedings of the 2013 IEEE 20th International Conference on Web Services, ICWS 2013*, pp. 75–82, USA, July 2013.

[19] H. Wang, Y. Tao, Q. Yu et al., "Incorporating both qualitative and quantitative preferences for service recommendation," *Journal of Parallel & Distributed Computing*, vol. 114, 2017.

[20] Y. Yu, H. Ma, and M. Zhang, "A Genetic Programming approach to distributed QoS-aware web service composition," in *Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014*, pp. 1840–1846, China, July 2014.

[21] Z. Ding, J. Liu, Y. Sun, C. Jiang, and M. Zhou, "A transaction and QoS-aware service selection approach based on genetic algorithm," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 7, pp. 1035–1046, 2015.

[22] M. Cremene, M. Suciu, D. Pallez, and D. Dumitrescu, "Comparative analysis of multi-objective evolutionary algorithms for QoS-aware web service composition," *Applied Soft Computing*, vol. 39, pp. 124–139, 2016.

[23] Q. Yu, L. Chen, and B. Li, "Ant colony optimization applied to web service compositions in cloud computing," *Computers & Electrical Engineering*, vol. 41, pp. 18–27, 2015.

[24] S. Bharathan, C. Rajendran, and R. P. Sundarraj, "Penalty based mathematical models for web service composition in a geo-distributed cloud environment," in *Proceedings of the 24th IEEE International Conference on Web Services, ICWS 2017*, pp. 886–889, USA, June 2017.

[25] W. Chen, I. Paik, and P. C. K. Hung, "Constructing a global social service network for better quality of web service discovery," *IEEE Transactions on Services Computing*, vol. 8, no. 2, pp. 284–298, 2015.

[26] S. Chakhar, "QoS-enhanced broker for composite web service selection," in *Proceedings of the 8th International Conference on*

Signal Image Technology and Internet Based Systems, SITIS 2012, pp. 533–540, Italy, November 2012.

[27] A. ShaikhAli, O. F. Rana, R. Al-Ali, and D. W. Walker, "UDDIe: An extended registry for Web services," in *Proceedings of the 2003 Symposium on Applications and the Internet Workshops, SAINT 2003*, pp. 85–89, USA, January 2003.

[28] L. Li, M. Rong, and G. Zhang, "A web service composition selection approach based on multi-dimension QoS," in *Proceedings of the 8th International Conference on Computer Science and Education, ICCSE 2013*, pp. 1463–1468, Sri Lanka, August 2013.

[29] L. Y. Qi, Y. Tang, W. C. Dou, and J. J. Chen, "Combining local optimization and enumeration for QoS-aware web service composition," in *Proceedings of the IEEE International Conference on Web Services*, pp. 34–41, July 2010.

[30] S. Wang, C.-H. Hsu, Z. Liang, Q. Sun, and F. Yang, "Multi-user web service selection based on multi-QoS prediction," *Information Systems Frontiers*, vol. 16, no. 1, pp. 143–152, 2014.

[31] M. Ortega, Y. Rui, K. Chakrabarti, S. Mehrotra, and T. S. Huang, "Supporting similarity queries in MARS," in *Proceedings of the 1997 5th ACM International Multimedia Conference*, pp. 403–413, November 1997.

[32] B. Hofreiter and S. Marchand-Maillet, "Rank aggregation for QoS-aware Web service selection and composition," in *Proceedings of the 6th IEEE International Conference on Service-Oriented Computing and Applications, SOCA 2013*, pp. 252–259, USA, December 2013.

[33] Y. Hao, "Enhanced resource scheduling in Grid considering overload of different attributes," *KSII Transactions on Internet and Information Systems*, vol. 10, no. 3, pp. 1071–1090, 2016.

[34] Y. Hao, M. Xia, N. Wen et al., "Parallel task scheduling under multi-clouds," *KSII Transactions on Internet and Information Systems*, vol. 11, no. 1, pp. 39–60, 2017.

Hindawi
Complexity
Volume 2019, Article ID 8258197, 7 pages
https://doi.org/10.1155/2019/8258197

Research Article

pSPARQL: A Querying Language for Probabilistic RDF Data

Hong Fang

College of Arts and Sciences, Shanghai Polytechnic University, Shanghai 201209, China

Correspondence should be addressed to Hong Fang; fanghong@sspu.edu.cn

Received 19 December 2018; Accepted 19 February 2019; Published 26 March 2019

Guest Editor: Jianxin Li

More and more linked data (taken as knowledge) can be automatically generated from nonstructured data such as text and image via learning, which are often uncertain in practice. On the other hand, most of the existing approaches to processing linked data are mainly designed for certain data. It becomes more and more important to process uncertain linked data in theoretical aspect. In this paper, we present a querying language framework for probabilistic RDF data (an important uncertain linked data), where each triple has a probability, called pSRARQL, built on SPARQL, recommended by W3C as a querying language for RDF databases. pSPARQL can support the full SPARQL and satisfies some important properties such as well-definedness, uniqueness, and some equivalences. Finally, we illustrate that pSPARQL is feasible in expressing practical queries in a real world.

1. Introduction

Resource Description Framework (RDF) [1] is the standard data model in the Semantic Web. In our real world, RDF data (as a knowledge base) possibly contains some uncertainty data due to the diversity of data sources, where RDF data are automatically extracted from different sources, such as YAGO [2]. For instance, some RDF data is generated from raw data via knowledge extraction and machine learning [3]. Indeed, uncertainty is generally a basic feature of data [4–6]. However, RDF model itself provides little support for uncertain data [7]. SPARQL [8], as a querying language for RDF data officially recommended by W3C [9], is unable to process uncertain data [4].

There are many approaches to processing probabilistic RDF [10]. Reference [4] proposes a probabilistic model for SQL over relational data. Reference [5] presents a Bayesian network to represent probabilistic relations in RDF. Reference [11] develops a framework for evaluating SPARQL conjunctive queries (i.e., basic graph patterns, BGP) on RDF probabilistic databases. Reference [12] proposes answering SPARQL queries with RDFS reasoning on probabilistic models that encode statistical relationships among correlated triples, where the proposed probability models are based on either probability distribution function or a disjunctive normal form probability problem. Reference [13] presents effective pruning mechanisms, as well as structural and probabilistic pruning for query answering of SPARQL conjunctive queries (i.e., BGP) over probabilistic RDF data graphs. Reference [14] presents a RESCAL-based approach to query processing in relational data via factorization. Reference [14] presents a heuristic algorithm for query answering of SPARQL conjunctive queries (i.e., BGP) over incomplete and uncertain RDF. Reference [15] presents a framework for SPARQL query answering over probabilistic databases by extending the rich semantics offered by ontologies with probabilistic information. Reference [16] presents a probabilistic knowledge base system, ARCHIMEDESONE, for query answering with inference by scaling up the knowledge expansion and statistical inference algorithms. Reference [17] proposes a probabilistic automata-based framework of query evaluation in the presence of uncertainty efficiently.

Although those approaches can query probabilistic RDF, most of them mainly process SPARQL conjunctive queries, that is, BGP queries. However, those existing probability models have little support for expressive operators (for instance, neither [13] nor [16] discusses OPTIONAL query for RDF) such as OPTIONAL, which is the least conventional operator of SPARQL [18], and DIFF, a difference operator in SPARQL 1.1 [19], which brings more expressivity [20].

In this paper, we present an extended querying language (called *pSPARQL: probabilistic SPARQL*) for probabilistic

RDF databases with support of the full SPARQL fragment. We show that the semantics of pSPARQL can satisfy some important properties such as well-definedness, uniqueness, and some equivalences. Compared with the previous poster in ISWC 2016 [21], in this paper, we present a totally new probabilistic representation model and prove that the newly proposed model can preserve some important properties such as uniqueness and distributive law of equivalence.

The remainder of this paper is structured as follows: the next section recalls RDF and SPARQL. Section 3 introduces the syntax and semantics of pSPARQL and Section 4 discusses some important properties. Finally, we summarize our work in the last section.

2. RDF and SPARQL

In this section, we briefly recall the syntax and semantics of SPARQL. For more readings, please refer to the core SPARQL formalization in [22].

2.1. RDF Graphs. Let I and L be infinite sets of *IRIs* and *literals*, respectively, with $I \cap L = \emptyset$. Let $U = I \cup L$. A triple $(s, p, o) \in U \times I \times U$ is called an *RDF triple*. An *RDF graph* is a finite set of RDF triples.

2.2. Syntax of SPARQL. Let V be a set of *variables*. SPARQL *patterns* are inductively defined as follows:

(i) Any triple from $(U \cup V) \times (I \cup V) \times (U \cup V)$ is a pattern (called a *triple pattern*).

(ii) If P_1 and P_2 are patterns, then so are the following: P_1 UNION P_2, P_1 AND P_2, P_1 DIFF P_2, and P_1 OPT P_2.

(iii) If P is a pattern and F is a constraint (defined next), then P FILTER F is a pattern; we call F the *filter*, which is a Boolean combination of *atomic constraints*, one of the three following forms: bound($?x$) (*bound*), $?x = ?y$ (*equality*), and $?x = c$ (constant equality), for $?x, ?y \in V$ and $c \in U$.

2.3. Semantics of SPARQL. Now, given a graph G and a pattern P, we define the semantics of P on G, denoted by $[\![P]\!]_G$, as a set of mappings (i.e., partial functions from V to U, in the following manner, where we use dom(μ) to denote the domain of μ)

(i) $[\![(u, v, w)]\!]_G := \{\mu : \{u, v, w\} \cap V \longrightarrow U \mid (\mu(u), \mu(v), \mu(w)) \in G\}$.

(ii) $[\![P_1 \text{ UNION } P_2]\!]_G := [\![P_1]\!]_G \cup [\![P_2]\!]_G$.

(iii) $[\![P_1 \text{ AND } P_2]\!]_G = \{\mu_1 \cup \mu_2 \mid \mu_1 \in [\![P_1]\!]_G \text{ and } \mu_2 \in [\![P_2]\!]_G \text{ and } \mu_1 \sim \mu_2\}$.

Here, two mappings μ_1 and μ_2 are called *compatible*, denoted by $\mu_1 \sim \mu_2$, if for any

$?x \in \text{dom}(\mu_1) \cap \text{dom}(\mu_2) = \emptyset, \mu_1(?x) = \mu_2(?x)$.

(iv) $[\![P_1 \text{ DIFF } P_2]\!]_G = \{\mu_1 \in [\![P_1]\!]_G | \neg\exists \mu_2 \in [\![P_2]\!]_G, \mu_1 \sim \mu_2\}$.

(v) $[\![P_1 \text{ OPT } P_2]\!]_G = [\![(P_1 \text{ AND } P_2) \text{ UNION } ((P_1 \text{ DIFF } P_2)]\!]_G$.

(vi) $[\![P_1 \text{ FILTER } P_2]\!]_G := \{\mu \in [\![P_1]\!]_G \mid \mu(F) = true\}$.

Here, for any mapping μ and filter F, the evaluation of F on μ, denoted by $\mu(F)$, is defined in terms of a three-valued logic with truth values *true*, *false*, and *error*. Recall that

F is a Boolean combination of atomic constraints.

For a bound constraint bound($?x$), we define

$$\mu(\text{bound}(?x)) = \begin{cases} true & if\ ?x \in \text{dom}(\mu); \\ false & otherwise. \end{cases} \tag{1}$$

For an equality constraint $?x = ?y$, we define

$$\mu(?x = ?y) = \begin{cases} true & if\ ?x, ?y \in \text{dom}(\mu)\ and\ \mu(?x) = \mu(?y); \\ false & if\ ?x, ?y \in \text{dom}(\mu)\ and\ \mu(?x) \neq \mu(?y); \\ error & otherwise. \end{cases} \tag{2}$$

Thus, when $?x$ and $?y$ do not both belong to dom(μ), the equality constraint evaluates to *error*. Similarly, for a constant-equality constraint $?x = c$, we define

$$\mu(?x = c) = \begin{cases} true & if\ ?x \in \text{dom}(\mu)\ and\ \mu(?x) = c; \\ false & if\ ?x \in \text{dom}(\mu)\ and\ \mu(?x) \neq c; \\ error & otherwise. \end{cases} \tag{3}$$

A Boolean combination is evaluated using the truth tables given in Table 1.

3. Probabilistic RDF and pSPARQL

In this section, we present probabilistic RDF and introduce the syntax and semantics of pSPARQL.

3.1. Probabilistic RDF. *A probabilistic RDF* R is a pair (G, ρ) where G is an RDF graph and ρ is a total function from $G \longrightarrow (0, 1]$. Intuitively speaking, ρ is a probability function mapping each triple to a probability.

For instance, let $R_{\text{John}} = (G, \rho)$ be a probabilistic RDF with $G = \{t_1, t_2, t_3\}$ and ρ is a function from $G \longrightarrow [0, 1]$ defined in Table 2.

Note that we assign a triple to a probability so that we could take triples as atoms in our scenario analogously treated in [13, 16]. This treatment is not direct to characterize the probability of subjects/objects in triples.

3.2. pSPARQL: A Probabilistic SPARQL. In this section, we introduce a probabilistic SPARQL (for short, pSPARQL).

The Syntax of pSPARQL. The syntax of pSPARQL is slightly different from the syntax of SPARQL [22] in filters, where we

newly introduce a fixed variable ?p to express constraints of probability.

A probabilistic atomic filter is one of the four following forms: $?p \geq s, ?p \leq s, ?p = s$ and $?p \neq s$, where $s \in (0, 1]$. The filter of pSPARQL is a Boolean combination of atomic filters and probabilistic atomic filters.

All patterns are called probabilistic patterns in pSPARQL.

The Semantics of pSPARQL. The semantics of probabilistic patterns are defined in terms of sets of pairs of the form (μ, p) (called a solution (with probability), denoted by τ), where μ is a solution of probabilistic patterns and $p \in (0, 1]$. Note that we only consider pairs of form (μ, p) where $p > 0$.

Now, given a probabilistic RDF graph $R = (G, \rho)$ and a probabilistic pattern P, we define the semantics of P on R, denoted by $[\![P]\!]_R$, as a set of solutions with probability, in the following manner:

$$\begin{aligned}
&[\![(u, v, w)]\!]_R \coloneqq \{(\mu, p) : \{u, v, w\} \cap V \\
&\quad \longrightarrow U \mid (\mu(u), \mu(v), \mu(w)) \in G \text{ and } p \\
&\quad = \rho(\mu(u), \mu(v), \mu(w))\}. \\
&[\![P_1 \text{ UNION } P_2]\!]_R \coloneqq \{(\mu, p) \mid (\mu, p_1) \\
&\quad \in [\![P_1]\!]_R \textit{ or } (\mu, p_2) \in [\![P_2]\!]_R \textit{ and } p \\
&\quad = \max\{p_1, p_2\}\}.
\end{aligned} \tag{4}$$

By default, we set $p = \max\{p\}$.

$$\begin{aligned}
&[\![P_1 \text{ AND } P_2]\!]_R \coloneqq \{\mu_1 \cup \mu_2, p \mid (\mu_i, p_i) \\
&\quad \in [\![P_i]\!]_R \ (i = 1, 2) \\
&\textit{and } p = \max_{\mu_1 \cup \mu_2 = \mu_i \cup \mu_j} \min\{p_i, p_j \mid (\mu_i, p_i) \\
&\quad \in [\![P_1]\!]_R \textit{ and } (\mu_j, p_j) \in [\![P_2]\!]_R \textit{ and } \mu_i \sim \mu_j\} \\
&[\![P_1 \text{ DIFF } P_2]\!]_R \coloneqq \{(\mu_1, p_1) \in [\![P_1]\!]_R \mid \neg\exists(\mu_2, p_2) \\
&\quad \in [\![P_2]\!]_R \text{ s.t. } \mu_1 \sim \mu_2\}. \\
&[\![P_1 \text{ OPT } P_2]\!]_R \\
&\quad = [\![(P_1 \text{ AND } P_2) \text{ UNION } (P_1 \text{ DIFF } P_2)]\!]_R. \\
&[\![P_1 \text{ FILTER } F]\!]_G \coloneqq \{\tau = (\mu, p) \in [\![P_1]\!]_G \mid \tau(F) \\
&\quad = \textit{true}\}.
\end{aligned} \tag{5}$$

(i) For a nonprobabilistic filter F, $\tau(F) = \mu(F)$.

(ii) For a Boolean combination F, $\tau(F) = \mu(F)$.

(iii) For a probabilistic filter $?p \geq s$, we define

$$\tau(?p \geq s) = \begin{cases} \textit{true} & \textit{if } p \geq s; \\ \textit{false} & \textit{otherwise}. \end{cases} \tag{6}$$

Table 1: Truth tables for the three-valued semantics.

(a)

p	q	$p \wedge q$	$p \vee q$
true	*true*	*true*	*true*
true	*false*	*false*	*true*
true	*error*	*error*	*true*
false	*true*	*false*	*true*
false	*false*	*false*	*false*
false	*error*	*false*	*error*
error	*true*	*error*	*true*
error	*false*	*false*	*error*
error	*error*	*error*	*error*

(b)

p	$\neg p$
true	*false*
false	*true*
error	*error*

(iv) For a probabilistic filter $?p \leq s$, we define

$$\tau(?p \leq s) = \begin{cases} \textit{true} & \textit{if } p \leq s; \\ \textit{false} & \textit{otherwise}. \end{cases} \tag{7}$$

(v) For a probabilistic filter $?p = s$, we define

$$\tau(?p = s) = \begin{cases} \textit{true} & \textit{if } p = s; \\ \textit{false} & \textit{otherwise}. \end{cases} \tag{8}$$

(vi) For a probabilistic filter $?p \neq s$, we define

$$\tau(?p \neq s) = \begin{cases} \textit{true} & \textit{if } p \neq s; \\ \textit{false} & \textit{otherwise}. \end{cases} \tag{9}$$

Example 1. Given a pattern $P = (?x, \textit{sufferFrom}, ?y)$FILTER$?\ p \geq 0.5$ (i.e., we query those persons who have suffered from some illness with probability over 0.5), we can compute that $[\![P]\!]_{R_{\text{John}}} = \{(\mu, p)\}$ where $\mu = \{?x \longrightarrow \text{John}, ?y \longrightarrow \textit{Schizophrenia}\}$ and $p = 0.84$. However, let $\tau = (\{?x \longrightarrow \text{John}, ?y \longrightarrow \textit{Schizophrenia}\}, 0.32)$, $\tau \in [\![P]\!]_{R_{\text{John}}}$ since $\tau(?p \geq 0.5) = \textit{false}$.

Example 2. Given a pattern $P = (?x, \textit{sufferFrom}, ?y)$ AND $(?x, \textit{Treatedby}, ?z)$ (i.e., we query those who have suffered from some illness and have been treated), we can compute that $[\![P]\!]_{R_{\text{John}}} = \{\tau_1, \tau_2\}$, where

(i) $\tau_1 = (\{?x \longrightarrow \text{John}, ?y \longrightarrow \textit{Schizophrenia}, ?z \longrightarrow \textit{Psychiatrist}\}, 0.32)$;

(ii) $\tau_2 = (\{?x \longrightarrow \text{John}, ?y \longrightarrow \textit{MentalDisorder}, ?z \longrightarrow \textit{Psychiatrist}\}, 0.84)$.

Table 2

No	Triple: (t)	Probability: ($\rho(t)$)
t_1	(*John*, *sufferedFrom*, *Schizophrenia*)	0.32
t_2	(*John*, *sufferedFrom*, *MentalDisorder*)	0.84
t_3	(*John*, Treatedby, *Psychiatrist*)	0.95

Example 3. Given a pattern P = (?x, *sufferFrom*, *Schizophrenia*)UNION (?x, *Treatedby*, *Psychiatrist*) (i.e., we query those who have suffered from schizophrenia or those who are treated by psychiatrists); we can compute that $[\![P]\!]_{R_{\text{John}}} = \{(?x \longrightarrow \text{John}, 0.95)\}\}$.

Note that *?p* is slightly different from variables where the value of *?p* is variable via probability computation, while the value of other variables is fixed. Moreover, we disallow the comparison of probability in filters.

4. Well-Definedness, Uniqueness, and Equivalence of pSPARQL

In this section, we discuss some important properties of pSPARQL.

Firstly, we introduce a property called well-definedness, which can ensure that the semantics of pSPARQL are well defined.

Proposition 4 (well-definedness). *For any pSPARQL pattern P, for any probabilistic RDF R, for any solution* $(\mu, p) \in [\![P]\!]_R$, *we have* $p \in (0, 1]$.

Proof. By induction on the structure of P,

if P is a triple pattern (u, v, w), then $p = ((\mu(u), \mu(v), \mu(w))) \in (0, 1]$;

if P is of the form P_1 UNION P_2, then let us discuss the three cases:

(i) if $(\mu, p) \in [\![P_1]\!]_R$ but $(\mu, p) \notin [\![P_2]\!]_R$, then $p \in (0, 1]$;

(ii) if $(\mu, p) \in [\![P_2]\!]_R$ but $(\mu, p) \notin [\![P_1]\!]_R$, then $p \in (0, 1]$;

(iii) if $(\mu, p) \in [\![P_1]\!]_R$ and $(\mu, p) \in [\![P_2]\!]_R$, then $p = \max\{p_1, p_2\} \in (0, 1]$ by induction.

If P is of the form P_1 AND P_2, then this claim holds by induction, since there exists some solution $(\mu_1, p_1) \in [\![P_1]\!]_R$ and some solution $(\mu_2, p_2) \in [\![P_2]\!]_R$ with $\mu_1 \sim \mu_2$ such that

$$\begin{aligned} &\mu = \mu_1 \cup \mu_2 \\ &\text{and } p \geq \min\{p_1, p_2\} \\ &\text{and } p = \max_{\mu_1 \cup \mu_2 = \mu_i \cup \mu_j} \min\{p_i, p_j \mid (\mu_i, p_i) \\ &\quad \in [\![P_1]\!]_R \text{ and } (\mu_j, p_j) \in [\![P_2]\!]_R \text{ and } \mu_i \sim \mu_j\} \\ &\quad \in (0, 1]. \end{aligned} \tag{10}$$

If P is of the form P_1 DIFF P_2 or P_1 FILTER F, then this claim holds by induction, since

$$[\![P]\!]_R \subseteq [\![P_1]\!]_R. \tag{11}$$

Finally, if P is of the form P_1 OPT P_2, then this claim holds by the cases of P_1 AND P_2, P_1 UNION P_2, and P_1 DIFF P_2 by induction. □

Proposition 5 (uniqueness). *For any pSPARQL pattern P, for any probabilistic RDF R, for any two solutions* $(\mu, p), (\mu', p') \in [\![P]\!]_R$, *if* $\mu = \mu'$, *then* $p = p'$.

Proof. By induction on the structure of P, we have the following.

If P is a triple pattern (u, v, w), then this claim directly holds by definition, since $p_1 = p_2 = \rho(\mu(u), \mu(v), \mu(w)))$.

If P is of the form P_1 UNION P_2, then let us discuss the three cases:

(i) If $(\mu, p) \in [\![P_1]\!]_R$ but $(\mu, p) \notin [\![P_2]\!]_R$, then this claim holds by induction.

(ii) If $(\mu, p) \in [\![P_2]\!]_R$ but $(\mu, p) \notin [\![P_1]\!]_R$, then this claim holds by induction.

(iii) If there exist $(\mu, p_1) \in [\![P_1]\!]_R$ and $(\mu, p_2) \in [\![P_2]\!]_R$, then this claim holds by induction, since $p = \max\{p_1, p_2\}$.

If P is of the form P_1 AND P_2, then this claim holds by induction, since there exists some solution $(\mu_1, p_1) \in [\![P_1]\!]_R$ and some solution $(\mu_2, p_2) \in [\![P_2]\!]_R$ with $\mu_1 \sim \mu_2$ such that $\mu = \mu_1 \cup \mu_2$ and $= \max_{\mu_1 \cup \mu_2 = \mu_i \cup \mu_j} \min\{p_i, p_j \mid (\mu_i, p_i) \in [\![P_1]\!]_R \text{ and } (\mu_j, p_j) \in [\![P_2]\!]_R \text{ and } \mu_i \sim \mu_j\}$. Therefore, p is unique.

If P is of the form P_1 DIFF P_2 or P_1 FILTER F, then this claim holds by induction, since

$$[\![P]\!]_R \subseteq [\![P_1]\!]_R. \tag{12}$$

Finally, we discuss the equivalence of patterns in pSPARQL. Let P and Q be two patterns in pSPARQL. We say that P is *equivalent* to Q, denoted by $P \equiv_p Q$, if for any probabilistic RDF R, $[\![P]\!]_R = [\![Q]\!]_R$. □

Next, we show that pSPARQL satisfies the distributive law of equivalence, which is proven to be important in SPARQL.

Proposition 6 (distributive law). *Let P_1, P_2, and P_3 be three patterns in pSPARQL and let F be a filter. The following holds:*

(1) (P_1 UNION P_2) FILTER $F \equiv_p$ (P_1 FILTER F) UNION (P_2 FILTER F);

(2) P_1 AND (P_2 UNION P_3) $\equiv_p$ (P_1 AND P_2) UNION (P_1 AND P_3);

(3) $(P_1$ UNION $P_2)$ AND $P_3 \equiv_p (P_1$ AND $P_3)$ UNION $(P_2$ AND $P_3)$;

(4) $(P_1$ UNION $P_2)$ DIFF $P_3 \equiv_p (P_1$ DIFF $P_3)$ UNION $(P_2$ DIFF $P_3)$;

(5) $(P_1$ UNION $P_2)$ OPT $P_3 \equiv_p (P_1$ OPT $P_3)$ UNION $(P_2$ OPT $P_3)$.

Proof (sketch). The first claim directly holds by the definition.

Now, we show the second item. Let R be a probabilistic RDF of the form (G, ρ). If $(\mu, p) \in [\![P_1$ AND $(P_2$UNION $P_3)]\!]_R$, then there must exist some solution $(\mu, p') \in [\![(P_1$ AND $P_2)$ UNION $(P_1$ AND $P_3)]\!]_R$. By Proposition 5, we can conclude that $p = p'$. Then $[\![P_1$ AND $(P_2$ UNION $P_3]\!]_R \subseteq [\![(P_1$ AND $P_2) \cup (P_1$ UNION $P_3]\!]_R$.

On the other hand, $(\mu, p) \in [\![(P_1$ AND $P_2) \cup (P_1$ UNION $P_3]\!]_R$; then there must exist some solution $(\mu, p') \in [\![P_1$ AND $(P_2$ UNION $P_3)]\!]_R$. By Proposition 5, we can conclude that $p = p'$. Then $[\![(P_1$ AND $P_2) \cup (P_1$ UNION $P_3]\!]_R \subseteq [\![P_1$ AND $(P_2$ UNION $P_3]\!]_R$.

Analogously, we can prove the third item and the fourth item.

Finally, we could prove the fifth item by using the third item and the fourth item. □

5. A Practical Example

In this section, we illustrate the application of pSPARQL in a real world via a practical example, where a probabilistic RDF is introduced in [11] shown in Figure 1.

Consider the following four queries (Q_1, Q_2, Q_3, Q_4) in pSPARQL.

(1) Q_1: What causes fatigue associated with some illness over 0.65 probability?

Q_1 is formally expressed in pSPARQL as follows:

SELECT *?x* ((*Fatigue*, *CauseOf*, *?x*) AND ((*?x*, *AssociatedWith*, *?z*) FILTER *?p* >0.65).

The solution of Q_1 is as follows:

$$[\![Q_1]\!]_{Rrsv} = \{(\{?x \longrightarrow Flu\}, 0.6)\}. \quad (13)$$

Note that $[\![(?x, AssociatedWith, ?z)]\!]_{Rrsv} = \{(\{?x \longrightarrow Flu, ?z \longrightarrow Cough\}, 0.7), (\{?x \longrightarrow Pneumonia, ?z \longrightarrow Bronchitis\}, 0.6)\}$. Thus $[\![(?x, AssociatedWith, ?z)$ FILTER$?p > 0.65]\!]_{Rrsv} = \{(\{?x \longrightarrow Flu, ?z \longrightarrow Cough\}, 0.7)\}$. Then $[\![Q_1]\!]_{Rrsv} = \{(\{?x \longrightarrow Flu\}, 0.6)\}$, since $[\![(Fatigue, CauseOf, ?x)]\!]_{Rrsv} = \{(\{?x \longrightarrow Flu\}, 0.6), (\{?x \longrightarrow Pneumonie\}, 0.6)\}$.

(2) Q_2: What are associated with cough over 0.7 probability?

Q_2 is formally expressed in pSRARQL as follows:

SELECT *?x* ((*?x*, *AssociatedWith*, *?y*) FILTER *?y* = '*Cough*' ∧*?p* >0.7).

The solution of Q_2 is as follows:

$$[\![Q_2]\!]_{Rrsv} = \{(\{?x \longrightarrow Bronchitis, ?y \longrightarrow Cough\}, 0.8)\}. \quad (14)$$

Table 3

?x	*?y*	*?p*
Bronchitis	*Cough*	0.8
Bronchitis	*RSV*	0.6
RSV	*Cough*	0.7
Flu	*Cough*	0.7
Pneumonia	*Bronchitis*	0.6

Note that $[\![(?x, AssociatedWith, ?y)]\!]_{Rrsv}$ is shown in Table 3.

Thus $[\![(?x, AssociatedWith, ?y)$ *FILTER*$? y =$ '*Cough*'$]\!]_{Rrsv} = \{(\{?x \longrightarrow$ Bronchitis, $?y \longrightarrow$ Cough$\}, 0.8), (\{?x \longrightarrow$ RSV, $?y \longrightarrow$ Cough$\}, 0.7), (\{?x \longrightarrow$ Flu, $?y \longrightarrow$ Cough$\}, 0.7)\}$.

Then $[\![Q_2]\!]_{Rrsv} = \{(\{?x \longrightarrow Bronchitis, ?y \longrightarrow Cough\}, 0.8)\}$.

(3) Q_3: What is probability of bronchitis associated with cough directly or indirectly?

Q_3 is formally expressed as follows:

SELECT *?x* ((*?x*, *AssociatedWith*, *?y*) UNION ((*?x*, *AssociatedWith*, *?z*) AND (*?z*, *AssociatedWith*, *?y*)) FILTER *?x* = 'Bronchitis'∧*?y* = '*Cough*').

The solution of Q_3 is as follows:

$$[\![Q_3]\!]_{R_{rsv}} = \{(\{?x \longrightarrow Bronchitis, ?y \longrightarrow Cough\}, 0.8)\}. \quad (15)$$

Note that $[\![((?x, AssociatedWith, ?y)]\!]_{R_{rsv}}$ is shown in Table 4.

Note that $[\![?x, AssociatedWith, ?z)$ AND (?z, $AssociatedWith, ?y)]\!]_{R_{rsv}}$ is shown in Table 5.

Thus $[\![(?x, AssociatedWith, ?y)$ UNION $((?x, AssociatedWith, ?z)$ AND $(?z, AssociatedWith, ?y))]\!]_{Rrsv}$ is shown in Table 6.

Then

$$[\![Q_3]\!]_{Rrsv} = \{(\{?x \longrightarrow Bronchitis, ?y \longrightarrow Cough\}, 0.8)\}. \quad (16)$$

(4) Q_4: What are associated with cough excluding bronchitis?

Q_4 is expressed in pSPARQL as follows:

SELECT *?x* (((*?x*, *AssociatedWith*, *?y*) FILTER *?y* = '*Cough*') DIFF ((*?x*, *AssociatedWith*, *?y*) FILTER *?x* = '*Bronchitis*' ∧*?y* = '*Cough*')).

The solution of Q_4 is as follows:

$$[\![Q_4]\!]_{Rrsv} = \{(\{?x \longrightarrow Flu, ?y \longrightarrow Cough\}, 0.7)\}. \quad (17)$$

Note that $[\![(?x, AssociatedWith, ?y)$FILTER$?\ y =$ '*Cough*'$]\!]_{R_{rsv}}$ is shown in Table 7.

Note that $[\![(?x, AssociatedWith, ?y)$FILTER$? x =' Bronchitis' \wedge ?y =$ '*Cough*'$]\!]_{R_{rsv}}$ is shown in Table 8.

Then $[\![Q_4]\!]_{Rrsv} = \{(\{?x \longrightarrow Flu, ?y \longrightarrow Cough\}, 0.7)\}$.

In short, we could express many interesting queries with respect to probabilistic RDF via pSPARQL, which are useful

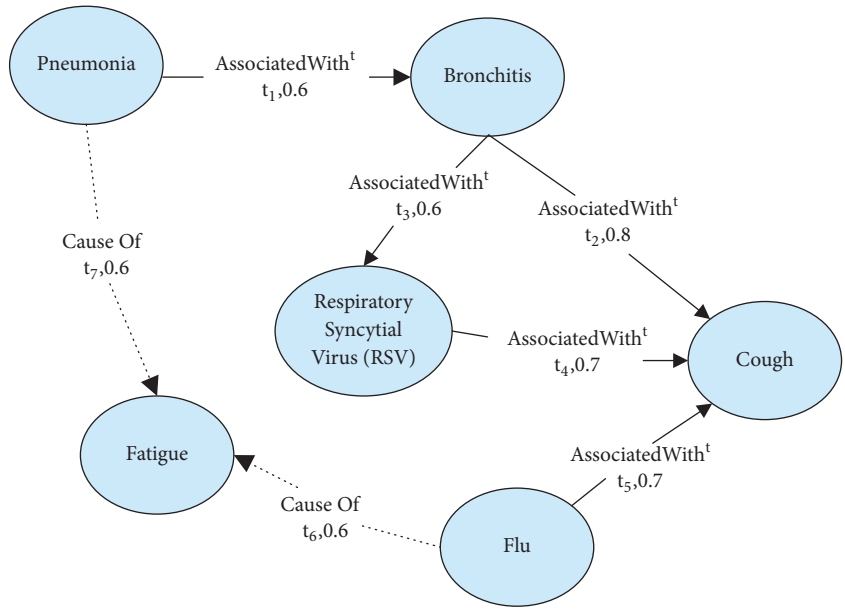


Figure 1: A virus RDF graph R_{rsv} [11].

Table 4

?x	*?y*	*?p*
Bronchitis	*Cough*	0.8
Bronchitis	*RSV*	0.6
RSV	*Cough*	0.7
Flu	*Cough*	0.7
Pneumonia	*Bronchitis*	0.6

Table 5

?x	*?y*	*?z*	*?p*
Bronchitis	*Cough*	*RSV*	0.6
Pneumonia	*Cough*	*Bronchitis*	0.6
Pneumonia	*RSV*	*Bronchitis*	0.6

Table 6

?x	*?y*	*?z*	*?p*
Bronchitis	*Cough*		0.8
Bronchitis	*RSV*		0.6
RSV	*Cough*		0.7
Flu	*Cough*		0.7
Pneumonia	*Bronchitis*		0.6
Bronchitis	*Cough*	*RSV*	0.6
Pneumonia	*Cough*	*Bronchitis*	0.6
Pneumonia	*RSV*	*Bronchitis*	0.6

in a practical world. Compared with SPARQL, where we obtain only connection via SPARQL querying, we could quantize the connection via pSPARQL, so that we could obtain more specific solutions.

6. Conclusions

In this paper, we extended SPARQL to support querying over probabilistic RDF. In the future, we will discuss some

Table 7

?x	?y	?p
Bronchitis	*Cough*	0.8
RSV	*Cough*	0.7
Flu	*Cough*	0.7

Table 8

?x	?y	?p
Bronchitis	*Cough*	0.8

foundational properties of pSPARQL and implement it in a prototype to provide the full SPARQL query answering services for probabilistic RDF. As a future work, we are interested in presenting probabilistic semantics of RDF graphs in a unified framework, where many applications could be supported.

Data Availability

No data were used to support this study.

Disclosure

An earlier version of this work was presented at "International Conference on Big Scientific Data Management 2018."

Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the program of the key discipline "Applied Mathematics" of Shanghai Polytechnic University (XXKPY1604).

References

[1] "RDF primer, W3C Recommendation, February 2004".

[2] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," in *Proceedings of the 16th International World Wide Web Conference (WWW '07)*, pp. 697–706, Alberta, Canada, May 2007.

[3] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases," *AI Magazine*, vol. 17, no. 3, pp. 37–53, 1996.

[4] N. Dalvi and D. Suciu, "Efficient query evaluation on probabilistic databases," in *Proceedings of the VLDB'04*, pp. 864–875, 2004.

[5] Y. Fukushige, "Representing probabilistic relations in RDF in," in *Proceedings of the ISWC-URSW'05*, pp. 106-107, 2005.

[6] D. Suciu, *Probabilistic Databases, Encyclopedia of Database Systems*, Springer, 2009.

[7] O. Udrea, V. Subrahmanian, and Z. Majkic, "Probabilistic RDF," in *Proceedings of the 2006 IEEE International Conference on Information Reuse & Integration*, pp. 172–177, Waikoloa Village, HI, USA, September 2006.

[8] "SPARQL query language for RDF, W3C Recommendation, January 2008".

[9] P. T. Wood, "Query languages for graph databases," *SIGMOD Record*, vol. 41, no. 1, pp. 50–60, 2012.

[10] A. Khan and L. Chen, "On uncertain graphs modeling and queries," in *Proceedings of the PVLDB Endowment*, vol. 8, pp. 2042-2043, 2015.

[11] H. Huang and C. Liu, "Query evaluation on probabilistic RDF databases," in *Proceedings of the WISE'09*, pp. 307–320, 2009.

[12] C. Szeto, E. Hung, and Y. Deng, "SPARQL query answering with RDFS reasoning on correlated probabilistic data," in *Proceedings of the WAIM'11*, pp. 56–67, 2011.

[13] X. Lian and L. Chen, "Efficient query answering in probabilistic RDF graphs," in *Proceedings of the the 2011 international conference*, p. 157, Athens, Greece, June 2011.

[14] D. Krompaß, M. Nickel, and V. Tresp, "Querying factorized probabilistic triple databases," in *Proceedings of the ISWC'14*, pp. 114–129, 2014.

[15] J. Schoenfisch, "Querying probabilistic ontologies with SPARQL," in *Proceedings of the KI'14*, pp. 2245–2256, 2014.

[16] X. Zhou, Y. Chen, and D. Z. Wang, "ArchimedesOne: Query processing over probabilistic knowledge bases," *Proceedings of the VLDB Endowment*, vol. 9, no. 13, pp. 1461–1464, 2016.

[17] T. Andronikos, A. Singh, K. Giannakis, and S. Sioutas, "Computing probabilistic queries in the presence of uncertainty via probabilistic automata," in *Proceedings of the ALGOCLOUD'17*, pp. 106–120, 2017.

[18] X. Zhang and J. Van den Bussche, "On the primitivity of operators in SPARQL," *Information Processing Letters*, vol. 114, no. 9, pp. 480–485, 2014.

[19] "SPARQL 1.1 query language, W3C Recommendation, March 2013".

[20] X. Zhang, J. Van den Bussche, K. Wang, and Z. Wang, "On the satisfiability problem of patterns in SPARQL 1.1," in *Proceedings of the AAAI'18*, pp. 2054–2061, 2018.

[21] H. Fang and X. Zhang, "pSPARQL: a querying language for probabilistic RDF (extended abstract)," in *Proceedings of the ISWC'16*, Posters, 2016.

[22] J. Pérez, M. Arenas, and C. Gutierrez, "Semantics and complexity of SPARQL," *ACM Transactions on Database Systems (TODS)*, vol. 34, no. 3, pp. 1–45, 2009.

Hindawi
Complexity
Volume 2019, Article ID 2926749, 11 pages
https://doi.org/10.1155/2019/2926749

Research Article

A Joint Deep Recommendation Framework for Location-Based Social Networks

Omer Tal and Yang Liu

Department of Physics and Computer Science, Wilfrid Laurier University, Waterloo, Ontario, Canada

Correspondence should be addressed to Yang Liu; yangliu@wlu.ca

Received 1 December 2018; Revised 11 February 2019; Accepted 5 March 2019; Published 19 March 2019

Guest Editor: Jiajie Xu

Location-based social networks, such as Yelp and Tripadvisor, which allow users to share experiences about visited locations with their friends, have gained increasing popularity in recent years. However, as more locations become available, the need for accurate systems able to present personalized suggestions arises. By providing such service, point-of-interest recommender systems have attracted much interest from different societies, leading to improved methods and techniques. Deep learning provides an exciting opportunity to further enhance these systems, by utilizing additional data to understand users' preferences better. In this work we propose *Textual and Contextual Embedding-based Neural Recommender* (TCENR), a deep framework that employs contextual data, such as users' social networks and locations' geo-spatial data, along with textual reviews. To make best use of these inputs, we utilize multiple types of deep neural networks that are best suited for each type of data. TCENR adopts the popular multilayer perceptrons to analyze historical activities in the system, while the learning of textual reviews is achieved using two variations of the suggested framework. One is based on convolutional neural networks to extract meaningful data from textual reviews, and the other employs recurrent neural networks. Our proposed network is evaluated over the Yelp dataset and found to outperform multiple state-of-the-art baselines in terms of accuracy, mean squared error, precision, and recall. In addition, we provide further insight into the design selections and hyperparameters of our recommender system, hoping to shed light on the benefit of deep learning for location-based social network recommendation.

1. Introduction

In today's age of information, it has become a prerequisite to have reliable data prior to making any decision. Preferably, we expect to obtain reviews from like-minded users before investing our time and money for any product or service. Location-based social networks (LBSN), such as Yelp, TripAdvisor, and Foursquare, provide such information about potential locations, while allowing users to connect with their friends and other users that have similar tastes. And indeed, these networks are gaining popularity. Yelp recently (https://www.yelp.ca/factsheet) reported having 72 million mobile active users every month, while TripAdvisor achieved no less than 390 million monthly users (https://www.tripadvisor.com/TripAdvisorInsights/w828). However, as LBSNs have more users with various preferences and additional locations are being added on a daily basis, it becomes time consuming to browse through all possibilities before finding an appropriate restaurant or venue to visit.

Point-of-interest (POI) recommendation, a subfield of recommender systems (RS), attempts to provide LBSN users with personalized suggestions. Properly exploited, it can save time and effort for the end user and encourages her to make future use in the LBSN both as a consumer and as a content provider. Collaborative filtering (CF) is probably the most prominent RS paradigm. It is based on the assumption that users who had resembling past activities will have similar future preferences.

While POI recommendation methods usually attempt to solve a similar problem as the standard recommender system, they are required to overcome additional challenges. First, data sparsity, commonly referred to as the cold start problem, is based on the fact that most users will only interact with a small fraction of the possible locations and vice-versa. When adopting methods of CF, it becomes harder to represent users and locations based on similar past activities, when the variation is high. This issue is worsened for users and

locations that have few or no past interactions known to the system, as available data is insufficient to fully capture their features. Second, in contrast to other recommendation scenarios, the decision whether to visit a location depends not only on the target user's preferences but on that of her friends [1]. They might share different interests that are unknown to the system, resulting in a more complex decision process for the RS to learn. Furthermore, attributes that relate to the location itself and are unknown to the system may have impact on the decision. For example, the availability of parking or convenient public transport, or the popularity of the area itself, can be the decisive factor for a user. These challenges proved the classic CF methods to be insufficient and prone to overfit the data as reported by previous works [2, 3].

A common approach to address sparsity while acknowledging the user's deriving factors is by utilizing contextual data, such as social networks [2, 4], geographical locations [5, 6], categories [1], and textual reviews [7, 8]. Deep neural networks have been enthusiastically adopted in recent years [2, 9, 10] to employ such complex contextual inputs, while being able to process the vast amount of available data. Different neural network architectures have been introduced to improve the recommendation performance, such as multilayer perceptrons (MLP) [3, 11, 12], convolutional neural networks (CNN) [13–15], and recurrent neural networks (RNN) [16–18]. These techniques were shown to highly benefit from the addition of contextual attributes. As more data becomes available, incorporating these features presents a promising opportunity to improve the personalized POI recommendation process, as will be demonstrated in this paper.

Although numerous works established the potential of recommender systems based on deep learning, most have focused on only a single type of neural network that best suited their given task. However, in this work, we intend to incorporate multiple types of neural networks, i.e., MLP, CNN, and RNN, to provide POI recommendation using various types of inputs. First we will describe our proposed method, *Textual and Contextual Embedding-based Neural Recommender* (TCENR), a framework that takes users' social network, locations' geographical data, and textual reviews along with historical activities, to provide personalized top-k POI recommendations. By optimizing MLP and CNN jointly, the proposed model will learn different aspects of the same user-location interaction in conjunction and therefore will be better suited to capture the underlying factors in the user's selection. We will then present an extension of TCENR, denoted as $\text{TCENR}_{\text{seq}}$, where we replace the CNN component with that of an RNN and attempt to learn user preferences and location attributes by treating the written reviews as a sequential input, rather than by focusing on their most important words. Although the proposed solution has been developed to provide recommendations for specific types of inputs, we claim it can be easily generalized to a framework able to support additional features.

The main contributions of our work are as follows:

(1) We present TCENR, a framework that jointly trains MLP and CNN to provide POI recommendations, as well as a variation, $\text{TCENR}_{\text{seq}}$, that performs the same task while adopting RNN instead of the CNN component.

(2) To the best of our knowledge, no work has been done in jointly training MLP and CNN for the task of POI recommendation using social networks, geographical locations, and natural language reviews as inputs.

(3) Evaluated over the Yelp dataset, our proposed frameworks were found to outperform seven state-of-the-art baselines in terms of accuracy, MSE, precision, and recall.

(4) By comparing the two alternatives to our suggested model, we provide insight into the impact gained by analyzing textual reviews as a secondary input to the common past interactions, as well as a comparison of CNN and RNN for the task of sentiment analysis in the same experimental settings.

(5) We further present comprehensive analysis over the most important hyper-parameters and design selections of our proposed networks, shedding some light over the different components of deep neural networks in the task of POI recommendation.

In the following section, we first lay the background and introduce related works to our model. In Section 3 we develop our proposed frameworks, which are evaluated and analyzed in Section 4. Finally, we summarize our work and introduce future work in Section 5.

2. Related Work

Incorporating additional data about users and items is a common strategy to provide meaningful recommendations and to mitigate the cold-start problem. In the area of POI recommender systems, where the data is highly sparse, such practices are essential.

2.1. Context-Based Recommender Systems. Location-based social networks are usually rich in contextual inputs which present various opportunities for data enrichment in RS. Such features include time [9], spatial location [19], user's social network [4], item's meta-data [1], photos [20], and demographics [11].

Contextual data is usually incorporated into RS either as part of the input or as a regularizing factor. Reference [11, 21] exploits the strengths of MLP-based networks in modeling complex relationships by concatenating multiple feature embeddings to the input before feeding it to a series of nonlinear layers. The final layer's output is then a representation of a user-item interaction, adjusted to the given context. Sequentiality is very often introduced to recommender systems by treating sequences of past user and item interactions in a timely manner. In the case of POI recommendations, it consists of feeding sequences of check-ins to RNN-based layers such as long-short term memory (LSTM) [22] or the more concise gated recurrent units (GRU) [23].

The use of spatial data is often done by dividing the input space into roles and regions. Assuming users' behavior varies when traveling far from home, previous works [5, 6] generated two profiles for each user, one to be used in her home region while another in more distant locations. A recent approach [24–26] attempts to divide the input space into geographical regions before incorporated into the model, often by hierarchical structures. Although methods based on regions and roles are able to better distinguish user behaviors in varied locations, they do not provide a personalized user representation and can ignore potential shifts of preferences from one region to another. For example, a user might prefer to visit a Starbucks location in different regions, close or far from home. Enriching geographical features with additional data is demonstrated in [25, 27] where the next location prediction is partially determined by past sequences of check-ins. However, these methods are not generic and cannot be extended to include additional features, derived from social networks or textual data.

Furthermore, in case of tasks in highly sparse environments, such as POI recommendation, adding user, or item specific inputs may diminish the model's ability to generalize. However, applying the same data as a regulating factor can enhance the model's performance and reduce over-fitting. Such has been done in [4], where the similarity between connected users in the social network was used to constrain a matrix factorization (MF) model. Reference [2] utilized social networks and geographical distances to enforce similar embeddings for users and locations, thus improving the model's ability to generalize for users and locations with few historical records.

2.2. Text-Based Recommendation. Since many websites encourage users to provide a written explanation to their numeric ratings, textual reviews are one of the most popular types of data to be integrated into RS. Previous works had adopted probabilistic-based approaches to alleviate the data sparsity problem using textual input [28–31]. By expressing each review as a bag of words, LDA-based models are able to extract topics which can be used to represent users' interests and locations' characteristics [5, 6, 25]. These probabilistic methods are usually successful in handling issues that standard CF approaches struggle with, such as out-of-town recommendations where similar users lack sufficient historical data. However, as demonstrated in recent works [17, 32], failing to preserve the original order of words and ignoring their semantic meaning prevent the successful modeling of a given review. On the other hand, an emerging trend of adopting neural networks over reviews allows such learning without the loss of data. These implementations can generally be categorized into RNN-based [17, 18] and CNN-based [14, 15] models.

RNN-based recommender systems usually rely on the sequential structure of sentences to learn their meaning. In [18], the words describing a target item are fed to a bidirectional RNN layer. Following the GRU architecture, the model utilizes an accumulated context from successive words to provide better representation of each word. To preserve and update the context of words, the recurrent model has to manage a large number of parameters. The problem becomes more prominent when adopting the popular LSTM paradigm as done in [17].

Following its success in the field of computer vision [33], CNN-based models are gaining popularity in other areas, such as textual modeling [34]. By employing a sliding window over a given document, such networks are able to represent different features found within the text by identifying relevant subsets of words. Reference [15] follows the standard CNN structure, comprised of an embedding to represent the semantic meaning of each word, a convolution layer for generating local features, and a max pooling operation to identify the most relevant factors. In [14], two CNNs are developed to represent the target user and item based on their reviews. The resulting vectors are regarded as the user and item representations, which are then fed to a nonlinear layer to learn their corresponding rating.

We claim that by jointly learning contextual and textual based deep models our proposed method will better exploit the strengths of collaborative filtering, while being more resilient to its shortcomings in sparse scenarios. This will be achieved by learning users' and locations' representations as similarities in direct interactions, along with the correlation in underlying features extracted from their written reviews. The notable work of [35] proposed JRL, a framework that similarly attempts to jointly optimize multiple models, where each is responsible for learning a unique perspective of the same task by focusing on different inputs. However, while JRL is a general framework, focused on extendability to many types of input, our proposed method is tailored for the task of POI recommendation.

3. The Neural Recommender

3.1. Neural Network Architecture. The following recommender system aims to improve the POI recommendation task by learning user-location interactions using two parallel neural networks, as shown in Figure 1. The context-based network, presented in the left part of the figure, is designed to model the user-POI preferences using social and geographical attributes and based on a multilayer perceptron structure [2, 3]. Shown in the right side of Figure 1, the convolutional neural network is responsible for the textual modeling unit [14]. It attempts to learn the same preference by analyzing the underlying meaning in users' and locations' reviews. Each of the two networks is based on modeling the user/POI input individually with regard to their shared interaction, defined in the merge layers.

3.1.1. Context-Based Network Layers. To better capture the complex relations between users and locations in LBSN, we chose to adopt the multilayer perceptron architecture. By stacking multiple layers of nonlinearities, MLP is capable of learning relevant latent factors of its inputs. It is first fed with user and location vectors of sizes N and M, where each input tuple $\langle u, p \rangle$ is transformed into sparse one-hot encoding representations. The two fully connected embedding layers, found on top of the input layer, project the sparse representations of users and locations into smaller and denser

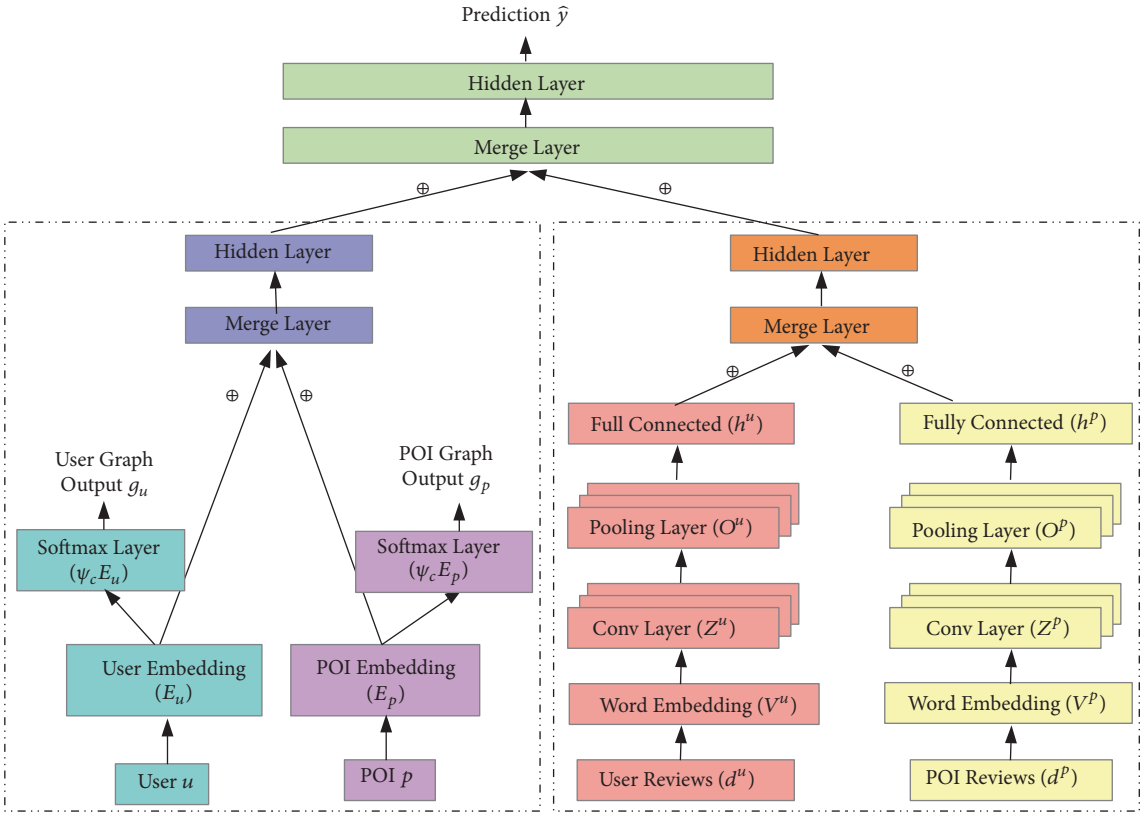


Figure 1: TCENR Framework.

vectors. For u and p, the respective embedding matrices are $E_u \in \mathbb{R}^{k_u \times N}$ and $E_p \in \mathbb{R}^{k_p \times M}$, where k_u and k_p are the corresponding dimensions.

We exploit the users' social networks along with the locations' geographical graphs to constrain the learned embeddings. Two softmax layers take the representations E_u and E_p as input and transform them back to N and M sized vectors, respectively. The user output layer, $\psi_c E_u \in \mathbb{R}^N$, describes the similarity of a user's embedding to all N users and can be formally denoted as

$$\psi_c E_u = a\left(W_c^u \times E_u + b_c^u\right), \tag{1}$$

where W_c^u and b_c^u are the layer's weight matrix and bias vector and a is a nonlinear activation function. Due to the similarity between the user and POI specific layers, the location output layer, $\psi_c E_p$, will not be developed in this section. Enforcing $\psi_c E_u$ and $\psi_c E_p$ to resemble user u's social graph, g_u, and location p's spatial graph, g_p, respectively, results in a smoothing factor that limits the amount by which connected entities' embeddings differ.

The user and location embeddings are projected to a merge layer and combined by a concatenation operator. Using concatenation instead of dot-product allows varied embedding structures, which in turn improves the generated representations [3]. As the input layer for the following neural network, the merge layer can be represented as $h^{(0)}(x)$, where

$$x = \left[E_u, E_p\right]. \tag{2}$$

Since simple concatenation of the user-location embedded vectors does not allow for interactions to be modeled, hidden layers are added to learn these connections. The popular Rectified Linear Units (ReLU) is employed as the activation function for these layers. More formally, the q-th hidden layer can be defined as

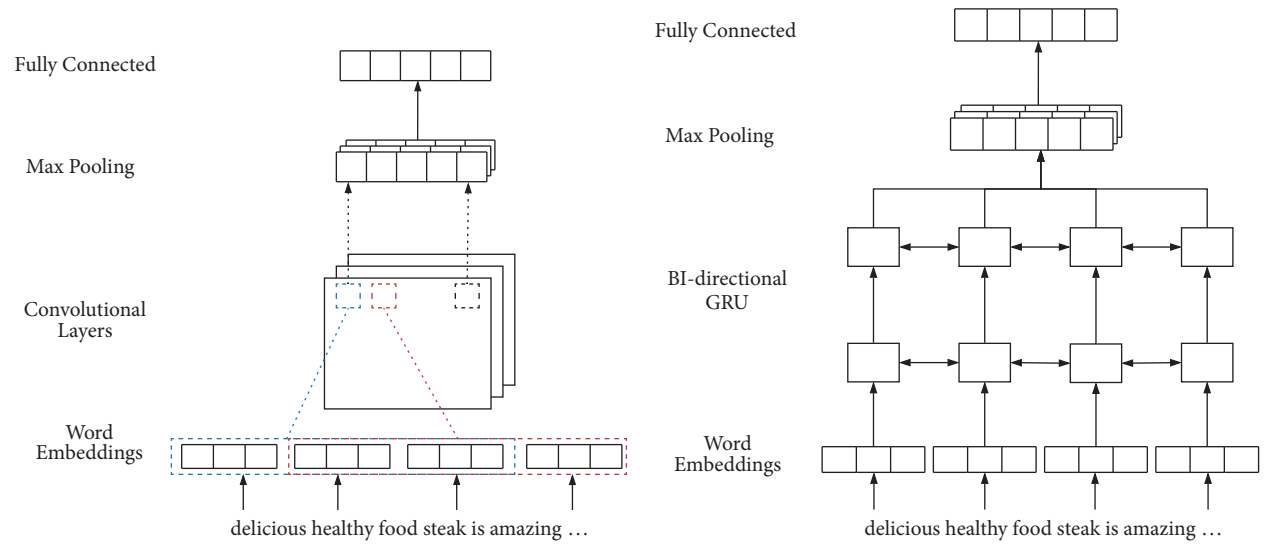
$$h_{context}^q(x) = ReLU\left(W^q h_{context}^{q-1}(x) + b^q\right), \tag{3}$$

where W^q and b^q are the q-th layer parameters.

3.1.2. Textual Modeling Network Layers. To improve the model's coverage, a textual-based network is introduced. It simultaneously learns the same interaction as the contextual-based network, but with a natural language input. Two additional vectors d^u and d^p, representing user u's and location p's textual reviews, respectively, are applied as inputs for this network. Each vector is comprised of all n words written by the user or about the location merged together, kept in their original order. These words are then mapped to c-dimensional vectors defining their semantic meaning in the following embedding layers. The output of the user embedding layer is the representation of all words used by a user u in the form of a matrix, and can be denoted as

$$V^u = \left[\phi\left(d_1^u\right), \ldots, \phi\left(d_l^u\right), \ldots, \phi\left(d_n^u\right)\right], \tag{4}$$

where $[\phi_1, \phi_2]$ denotes the concatenation of two vectors, d_l^u is user u's l-th word, and $\phi : D \longrightarrow \mathbb{R}^{k_w}$ is a lookup function to



(a) Textual modeling component using CNN

(b) Textual modeling component using RNN

FIGURE 2: Proposed alternatives to learn user and location representations from textual reviews. 2(a) is a CNN-based solution employed in TCENR, while 2(b) illustrates the suggested extension using RNN.

a pre-trained textual embedding layer [36, 37] that represents each word in vocabulary D as a vector in size k_w. Similarly, V^p denotes the word embedding matrix for location p.

Due to the large amount of parameters required to train the aforementioned contextual model, the textual network is implemented using a CNN-based architecture which is usually more computationally efficient than RNN. The semantic representations of users' and locations' reviews are fed to convolution layers, to detect parts of the text that best capture the review's meaning. These layers produce t feature maps over the embedded word vector, using a window size of ws and filter $K \in \mathbb{R}^{k_w \times t}$. As suggested by [14], ReLU is used as an activation function for this layer:

$$\begin{aligned} z_{jl}^u &= ReLU\left(V_{l:l+ws}^u * K_j^u + b_j^u\right), \\ z_j^u &= \left[z_{j1}^u, \ldots, z_{jl}^u, \ldots, z_{jn}^u\right], \end{aligned} \tag{5}$$

where V_l^u is user u's l-th input word embedding and z_j^u the j-th feature, extracted from the complete text.

Based on the standard CNN structure, feature maps produced by the convolution layers are reduced by a pooling layer:

$$\begin{aligned} o_j^u &= \max\left(z_j^u\right), \\ O^u &= \left[o_1^u, \ldots, o_j^u, \ldots, o_t^u\right], \end{aligned} \tag{6}$$

where max-pooling is selected to identify the most important words and their latent values. O^u is the collection of all concise features extracted from user u's textual input. These are followed by fully connected layers to jointly model the different features and result in the latent textual representations for user u and location p, respectively, denoted as h^u and h^p:

$$h^u = ReLU\left(W_1^u \times O^u + b_1^u\right). \tag{7}$$

To combine the outputs of the users and locations fully connected layers to the same feature space, a shared layer is utilized. It concatenates its two inputs and learns their interaction by employing an additional hidden layer:

$$h_{reviews} = ReLU\left(W_2 \times \left[h^u, h^p\right] + b_2\right). \tag{8}$$

The two neural networks are then finally merged to produce a prediction $\hat{y}_{up} \in [0, 1]$. The last layers of the two networks, each representing a different view of the user-location interaction, are concatenated and fed to yet another hidden layer, responsible to blend the learning:

$$\hat{y}_{up} = \sigma\left(W_3 \times \left[h_{context}, h_{reviews}\right] + b_3\right), \tag{9}$$

where the sigmoid function was selected to transform the hidden layer output to the desired range of $[0, 1]$.

3.2. Sequential Textual Modeling. To further investigate the gain achieved by integrating a textual modeling component over reviews in TCENR, we suggest an extension, denoted as $\text{TCENR}_{\text{seq}}$. Following its success in previous language modeling tasks [17, 18] and its ability to capture sentences' sequential nature, we employ an RNN component to learn latent features from reviews. An illustration of the proposed extension is presented in Figure 2(b), while the CNN method used in the vanilla TCENR is shown in Figure 2(a), to provide a convenient base for comparisons. More specifically we follow the findings of previous works [18, 23] and implement our recurrent network using GRU, an architecture that achieves competitive performance compared to LSTM, but with fewer parameters, making it more efficient:

$$\begin{aligned} f_l &= \sigma\left(W_f V_l^u + R_f h_{l-1} + b_f\right) \\ s_l &= \sigma\left(W_s V_l^u + R_s h_{l-1} + b_s\right) \end{aligned}$$

$$\begin{aligned}\widetilde{c}_l &= \tanh\left(W_{\widetilde{c}} V_l^u + f_l \odot R_{\widetilde{c}} h_{l-1} + b_{\widetilde{c}}\right)\\ h_l &= \left(1 - s_l\right) \odot h_{l-1} + s_l \odot \widetilde{c}_l,\end{aligned} \tag{10}$$

where f_l is the forget gate for input word l, s_l is the output gate, $\widetilde{c}_l$ is the new candidate state combining the current word embedding, V_l^u, with the previous hidden state, and h_l is current state for word l modeled by the output gate. $\odot$ denotes the element-wise product, and W_f, R_f, W_s, R_s, $W_{\widetilde{c}}$, and $R_{\widetilde{c}}$ are the GRU weight matrices while b_f, b_s, and $b_{\widetilde{c}}$ are the bias vectors.

Since the context of a word can be determined by other preceding and successive words or sentences, our proposed method employs a bidirectional GRU over the user embedding, V^u, and the location, V^p. Each word l's hidden state is learned by forward and backward GRU layers, denoted as $\overrightarrow{h_l^1}$ and $\overleftarrow{h_l^1}$, respectively. To learn a more concise and combined representation of a word while taking into account the context of all surrounding words, we feed the concatenation of $\overrightarrow{h_l^1}$ and $\overleftarrow{h_l^1}$ to an additional bidirectional GRU layer, such that its input for every word l is $e_l^2 = [\overrightarrow{h_l^1}, \overleftarrow{h_l^1}]$. The second recurrent unit will output n latent vectors, each is a sequentially infused representation of a word written by the target user or about the candidate item. To allow the method of textual modeling to be the only variant between TCENR and $\text{TCENR}_{\text{seq}}$ and to further reduce the number of learned parameters, all modified word vectors will be fed to the pooling and fully connected layers, originally presented in (6) and (7), respectively. This will allow us to directly determine the effect RNN has on textual modeling for POI recommender systems compared to CNN, as well as enabling the model to learn a more concise user and location representations. As in TCENR, the resulting vectors will be merged in order to learn the user-location interaction.

3.3. Training the Network. To train the recommendation models, we adopt a pointwise loss objective function, as done in [2, 3, 14], where the difference between the prediction $\widehat{y}_{up}$ and the actual value y_{up} is minimized. To address the implicit feedback nature of LBSNs, we sample a set of negative samples from the dataset, denoted as Y^-.

Due to the implicit feedback nature of the recommendation task, the algorithm's output can be considered as a binary classification problem. As the sigmoid activation function is being used over the last hidden layer, the output probability can be defined as

$$\begin{aligned}&p\left(Y, Y^- \mid E_u, E_p, V^u, V^p, \Theta_f\right)\\ &\quad= \prod_{(u,p)\in Y} \widehat{y}_{up} \prod_{u,p'\in Y^-} \left(1 - \widehat{y}_{up'}\right),\end{aligned} \tag{11}$$

where E_u and E_p are the embedding layers for users and locations, respectively. Similarly, V^u and V^p are the textual reviews embedding layers and Θ_f represents the model parameters. Taking the negative log-likelihood of p results in the binary cross-entropy loss function for the prediction portion of the model:

$$\begin{aligned}L_{pred} = &-\sum_{(u,p)\in Y\cup Y^-} y_{up} \log \widehat{y}_{up}\\ &+ \left(1 - y_{up}\right) \log\left(1 - \widehat{y}_{up}\right).\end{aligned} \tag{12}$$

As there are two more outputs in the model, the users' social network $\psi_c E_u$ and the locations' distance graph, $\psi_c E_p$, two additional loss functions are required to train the network. We follow the process done in [2], assuming two users who share the same context should have similar embeddings. This is achieved by minimizing the log-loss of the context given the instance embedding:

$$\begin{aligned}&L_{u\_context}\\ &\quad= -\sum_{(u,u_c)} \log\left(\psi_c E_u - \log \sum_{u_c'\in C_u} \exp\left(\psi_{c'} E_u\right)\right),\end{aligned} \tag{13}$$

where $\psi_c E_u$ is as defined in (1). Taking the binary class label into account prompts the following loss function, corresponding with minimizing the cross-entropy loss of user i and context c with respect to the y class label:

$$\begin{aligned}L_{u\_context} = &-I\left(y \in Y\right) \log \sigma\left(\psi_c E_u\right)\\ &- I\left(y \in Y^-\right) \log \sigma\left(-\psi_c E_u\right),\end{aligned} \tag{14}$$

where I is a function that returns 1 if y is in the given set, and 0 otherwise. The same logic is used to formulate the loss function for the POI context and will not be provided due to space limitations.

We simultaneously minimize the three loss functions L_{pred}, $L_{u\_context}$, and $L_{p\_context}$. The joint optimization improves the recommendation accuracy while enforcing similar representations for locations in close proximity and users connected in the social network. The loss functions are combined using two hyper-parameters, to weight the contextual contribution:

$$L = L_{pred} + \lambda_1 L_{u\_context} + \lambda_2 L_{p\_context}. \tag{15}$$

To optimize the combined loss function, a method of gradient descent can be adopted, and more specifically we utilize the Adaptive Moment Estimation (Adam) [38]. This optimizer automatically adjusts the learning rate and yields faster convergence than the standard gradient descent in addition to making the learning rate optimization process more efficient. In order to avoid additional overfitting when training the model, an early stopping criteria is integrated. The model parameters are initialized with Gaussian distribution, while the output layer's parameters are set to follow uniform distribution.

4. Experiments and Evaluation

4.1. Experimental Setup. To evaluate our proposed algorithm, we use Yelp's real-world dataset (https://www.yelp.com/dataset/challenge). It includes a subset of textual reviews along

with the users' friends and the businesses locations. Due to the limited resources used in the model evaluation, we chose to filter the dataset and keep only a concise subset, where all users and locations with less than 100 written reviews or less than 10 friends are removed. The filtered dataset includes 141,028 reviews and 98.08% sparsity for the rating matrix. The social and geographical graphs were constructed by random walks. 10% of the original vertices were sampled as base nodes, while 20 and 30 vertices were connected to each base node for users and locations, respectively, with a window size of 3. To build the POI graph, two locations are considered directly connected if they are up to 1 km apart.

To test our models' performance, the original data was split to training-validation-test sets by random sampling, with the respective ratios of 56%-24%-20%, resulting in 78,899 training instances. In addition, the input data was negatively sampled with 4 negative locations for every positive one.

To effectively compare our proposal with other alternatives, we adopt the same settings as applied in [2, 3]. The MLP input vectors are represented with an embedding size of 10, while two hidden layers are added on top of the merged result. Following the tower architecture, where the size of each layer is half the size of its predecessor, the numbers of hidden units are 32 and 16 for the first and second layers, respectively.

In the CNN component each word is represented by a pretrained embedding layer with 50 units, while the convolutional layer is constructed with a window size of 10 and a stride of 3. It results in 3 feature maps that are flattened after performing the max-pooling operation with a pool size of 2. The results are further modeled by a hidden layer with 32 units. Following the merge of the two hidden units, their interaction is learned using another hidden layer with 8 units. To combine the three loss functions as described in (15), we follow the results of [2] and set the hyperparameters $\lambda_1 = \lambda_2 = 0.1$. For the training phase of the model, a learning rate of 0.005 was used over 50 maximum epochs and a batch size of 512 samples.

4.2. Baselines. To evaluate our algorithm, we chose to compare it to these seven, empirically proven, frameworks:

(i) HPF [39]. Hierarchical Poisson matrix Factorization. A Bayesian framework for modeling implicit data using Poisson Factorization.

(ii) NMF [40]: Nonnegative Matrix Factorization, a CF method that takes only the rating matrix as input.

(iii) Geo-SAGE [5]: A generative method that predicts user check-ins in LBSNs using geographical data and crowd behaviors.

(iv) LCARS [6]: Location Content Aware Recommender System. A probabilistic model that exploits local preferences in LBSN and content information about POIs.

(v) NeuMF [3]. Neural Matrix Factorization. A state-of-the-art model combining MF with MLP on implicit ratings.

Table 1: Performance comparison over the Yelp dataset. Improvement of TCENR compared to each method is shown in brackets.

Model	Accuracy	MSE	Pre@10	Rec@10
HPF	0.8141	0.1800	0.5526	0.3699
	(1.69%)	(34.94%)	(18.51%)	(40.98%)
NMF	0.8222	0.1189	*0.7851*	0.3517
	(0.69%)	(1.51%)	*(-16.58%)*	(48.28%)
Geo-SAGE	0.7995	0.1807	0.2912	0.4145
	(3.55%)	(35.19%)	(124.89%)	(25.81%)
LCARS	0.8142	0.1612	0.6408	0.5127
	(1.68%)	(27.36%)	(2.2%)	(1.72%)
NeuMF	0.8273	0.1421	0.6488	*0.5586*
	(0.07%)	(17.59%)	(0.94%)	*(-6.64%)*
Pace	0.8239	0.1186	0.6406	0.5049
	(0.49%)	(1.26%)	(2.23%)	(3.29%)
DeepCoNN	0.8037	0.1454	0.5385	0.323
	(3.01%)	(19.46%)	(21.62%)	(64.46%)
TCENR	*0.8279*	0.1171	0.6549	*0.5215*
$\text{TCENR}_{\text{seq}}$	0.8273	*0.1161*	*0.6655*	0.4738
	(0.07%)	*(-0.86%)*	*(-1.59%)*	(10.07%)

(vi) PACE [2]. Preference and Context Embedding. A MLP-based framework with the addition of contextual graphs' smoothing for POI recommendation.

(vii) DeepCoNN [14]. Deep Cooperative Neural Networks. A CNN-based method that jointly learns an explicit prediction by exploiting users' and locations' natural language reviews.

For the task of evaluating our model and the baselines, we chose to apply Accuracy and Mean Square Error (MSE) over all n test samples, as well as Precision (Pre@10) and Recall (Rec@10) for the average top 10 predictions per user.

The proposed models were implemented using Keras (https://keras.io) on top of TensorFlow (https://www.tensorflow.org) backend. All experiments were conducted using Nivida GTX 1070 GPU.

4.3. Performance Evaluation. The performance of our proposed algorithms and the seven baselines is reported in Table 1, along with the improvement ratio of TCENR over each method in brackets. The presented results are based on the average of three individual executions.

As can be witnessed from the results, the proposed model, TCENR, achieves the best results overall compared to all baselines. Furthermore, it was found to significantly outperform HPF, NMF, Geo-SAGE, LCARS, Pace, and DeepCoNN for $p < 0.05$ based on a one-sided unpaired t-test in terms of accuracy and MSE. The contrasting results in terms of precision and recall compared to NeuMF suggest that TCENR offers less,but more relevant recommendations to the user. While NMF provides the best precision score compared to all methods, it underperforms in all other measures, making it a less desirable model. Taking a closer look shows that, surprisingly, NeuMF outperforms PACE in accuracy, precision and recall. This may be due to the less

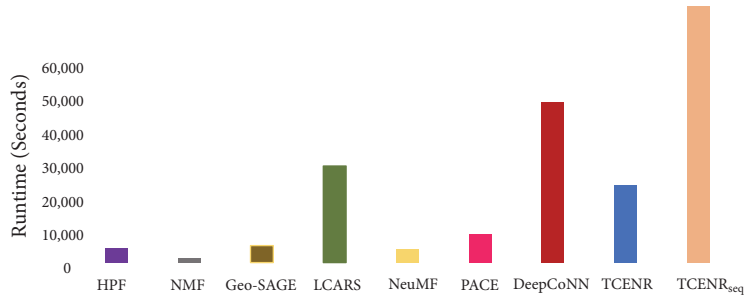


Figure 3: Runtime (seconds) of all models on the Yelp dataset.

sparse dataset tested, which does not allow the contextual regularization to be fully harvested. In addition, the use of only the first 500 words to represent the textual input for each user and location may explain the relatively low scores of the DeepCoNN model on the dataset, while the performance of Geo-SAGE and LCARS demonstrates that relying solely on geographical data does not allow such models to fully capture users' preferences in LBSNs.

Comparing TCENR and its proposed extension TCENR$_{seq}$ provides contrasting results. By employing RNN instead of CNN to extract user and location features from textual reviews, TCENR$_{seq}$ achieves lower error rate and improved precision score, while accuracy and recall are worsened. It may be considered that, by accurately capturing different aspects from user reviews, the model is able to reinforce its hypotheses and therefore reduce the uncertainty in some cases. However, when faced with a contrast between textual aspects and the ground truth, it might choose the wrong class label. Nonetheless, the results demonstrate the importance of adopting the most suitable techniques and measures to learn different data types, rather than employing a single method over all inputs. Moreover, it shows the positive impact of using textual data in conjunction to historical activities. The reported performance further suggests additional insight towards the selection of CNN and RNN for the task of language modeling in future recommendation tasks.

To further evaluate our suggested frameworks and the seven baselines in terms of runtime, the average time required to fully train each method is presented in Figure 3. As demonstrated by the results, TCENR is competitive with most baselines, and found to be more efficient than DeepCoNN and LCARS. The reported runtime of TCENR$_{seq}$ further demonstrates the relative efficiency of CNN-based solutions for textual modeling tasks. As the number of trainable parameters is increased due to the use of recurrent layers, our RNN-based extension takes 329% longer to train compared to TCENR, while achieving comparative results.

4.4. Model Design Analysis. In this section we discuss the effect of several design selections over the suggested model's performance.

4.4.1. Merge Layer. The importance of the model's final layers, responsible for combining the dense output of both the MLP and convolutional networks, requires a close attention, as it effects the networks' ability to jointly learn and the prediction itself. To properly select the fusion operator, the following methods had been considered:

(i) Combining the last hidden layers of the two models using concatenation. A model using this method will be denoted as $TCENR_{con}$ and described in (9).

(ii) Merging the last hidden layers using dot product, resulting in a model named $TCENR_{dot}$ that can be defined as

$$\hat{y}_{up} = h_{context} \cdot h_{reviews}. \tag{16}$$

(iii) Combining the two previously described methods, where the two representations will be jointly learned by concatenation and dot product. The resulted model will be denoted as $TCENR_{dot\_con}$ and can be developed by combining (9) and (16) using addition and translating the result to a range of $[0, 1]$ with the sigmoid function:

$$\hat{y}_{up} = \sigma\left(\sigma\left(W_4 \times \left[h_{context}, h_{reviews}\right] + b_4\right) + h_{context} \cdot h_{reviews}\right). \tag{17}$$

(iv) Adopting a weighted average for the prediction result of the two networks. Denoted as $TCENR_{weight}$, this model can be defined as

$$\hat{y}_{up} = \lambda_1 \sigma\left(W_5 \times h_{context} + b_5\right) + \lambda_2 \sigma\left(W_6 \times h_{reviews} + b_6\right). \tag{18}$$

As shown in Figure 4, adopting the more simple methods of weighted average and dot product leads to an inferior performance of TCENR, demonstrating the added value of utilizing the latent features learned by each subnetwork jointly. When combined with the underperforming method of dot product in $TCENR_{dot\_con}$, the use of concatenation improves over dot product alone. However, since the two methods are integrated using a simple average, employing only concatenation as done in $TCENR_{con}$ produces the best results, and therefore integrated into the final model.

4.4.2. MLP Layer Design. Although it was found by [3] that adding more layers and units to the MLP-based recommender has a positive effect, the use of CNN and the additional hidden layer suggests it is a subject worth investigating.

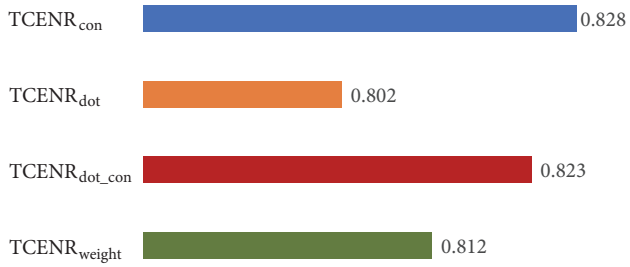


Figure 4: Comparison of merging methods in terms of accuracy.

Table 2: Model's accuracy with different layers.

1st layer	H=1	H=2	H=3	H=4
16	0.824	0.827	-	-
32	0.823	*0.837*	0.825	-
64	0.822	0.829	0.83	0.827
128	0.823	0.828	0.829	0.827

To this end, we test the proposed algorithm with 1-4 hidden layers used to learn the user-item interaction with contextual regularization in varying sizes from 8 to 128 hidden units. The results in terms of test set's accuracy are presented in Table 2, where the number of hidden layers is defined as columns and the size of the first unit is presented as rows. Unlike previous results, we find that two hidden layers with 32 and 16 hidden units result in the best performance for our dataset.

4.4.3. Number of Words. The use of written reviews in their original order allows the strengths of CNN and RNN to be exploited by finding the best representation for every few words and eventually for the whole text. Our final dataset, however, is composed of very long reviews, where to fully learn a single user or location, more than 20,000 words are required, making it computationally expensive to extract relevant representations. To benefit from the sequential nature of the written reviews while keeping the solution feasible, the number of words was limited to a range of 500-6000. As can be witnessed from Figure 5, there is a slight improvement in accuracy as the number of words increase up to 3000, while additional words result in an increased bias towards users and locations with longer reviews and in turn reduce the model's learning capabilities.

5. Conclusion and Future Work

In this paper, we developed a neural POI recommender system called TCENR. The model exploits data about users, locations, spatial data, social networks, and textual reviews to predict the implicit preference of users regarding POIs. TCENR models two types of user-location interactions: native check-ins regularized by contextual information and the words used to describe the users' experiences. We further extended our proposed method and presented $TCENR_{seq}$, where textual data was modeled using RNN instead of CNN. Evaluated over the Yelp dataset, the proposed algorithms consistently achieved superior results compared to seven state-of-the-art baselines in terms of accuracy and MSE.

For future work, we intend to extend our models' evaluation over additional LBSN datasets. In addition we plan to investigate the proposed frameworks' contribution to the cold-start problem by analyzing its performance on additional data, while taking new users and locations with few reviews into account.

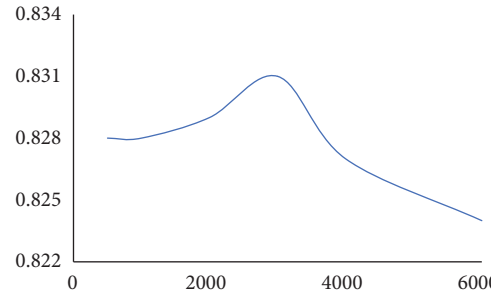


Figure 5: Number of words comparison in terms of accuracy.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China Grant (61572289) and NSERC Discovery Grants.

References

[1] H. Li, Y. Ge, R. Hong, and H. Zhu, "Point-of-interest recommendations: learning potential check-ins from friends," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery And Data Mining*, pp. 975–984, ACM, San Francisco, California, USA, August 2016.

[2] C. Yang, L. Bai, C. Zhang, Q. Yuan, and J. Han, "Bridging collaborative filtering and semi-supervised learning: a neural approach for poi recommendation," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1245–1254, ACM, Halifax, NS, Canada, August 2017.

[3] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web*, International World Wide Web Conferences Steering Committee, pp. 173–182, Perth, Australia, April 2017.

[4] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, "Recommender systems with social regularization," in *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*, pp. 287–296, ACM, February 2011.

[5] W. Wang, H. Yin, L. Chen, Y. Sun, S. Sadiq, and X. Zhou, "Geo-sage: a geographical sparse additive generative model for spatial item recommendation," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1255–1264, ACM, Sydney, NSW, Australia, August 2015.

[6] H. Yin, Y. Sun, B. Cui, Z. Hu, and L. Chen, "Lcars: a location-content-aware recommender system," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 221–229, ACM, Chicago, Illinois, USA, August 2013.

[7] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '11)*, pp. 448–456, ACM, August 2011.

[8] H. Wang, N. Wang, and D.-Y. Yeung, "Collaborative deep learning for recommender systems," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1235–1244, ACM, Sydney, NSW, Australia, August 2015.

[9] J. Manotumruksa, C. Macdonald, and I. Ounis, "A deep recurrent collaborative filtering framework for venue recommendation," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 1429–1438, ACM, Singapore, Singapore, November 2017.

[10] Q. Liu, S. Wu, D. Wang, Z. Li, and L. Wang, "Context-aware sequential recommendation," in *Proceedings of the 2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 1053–1058, IEEE, Barcelona, Spain, December 2016.

[11] H.-T. Cheng, L. Koc, J. Harmsen et al., "Wide & deep learning for recommender systems," in *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, pp. 7–10, ACM, 2016.

[12] Y. Yu, L. Zhang, C. Wang, R. Gao, W. Zhao, and J. Jiang, "Neural personalized ranking via poisson factor model for item recommendation," *Complexity*, vol. 2019, Article ID 3563674, 16 pages, 2019.

[13] A. Van Den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," in *Proceedings of the 26th International Conference on Neural Information Processing Systems*, Advances in neural information processing systems, pp. 2643–2651, 2013.

[14] L. Zheng, V. Noroozi, and P. S. Yu, "Joint deep modeling of users and items using reviews for recommendation," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 425–434, ACM, Cambridge, UK, Feburary 2017.

[15] D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, "Convolutional matrix factorization for document context-aware recommendation," in *Proceedings of the 10th ACM Conference on Recommender Systems*, pp. 233–240, ACM, 2016.

[16] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," 2015, https://arxiv.org/abs/1511.06939.

[17] A. Almahairi, K. Kastner, K. Cho, and A. Courville, "Learning distributed representations from reviews for collaborative filtering," in *Proceedings of the 9th ACM Conference on Recommender Systems*, pp. 147–154, ACM, Vienna, Austria, September 2015.

[18] T. Bansal, D. Belanger, and A. McCallum, "Ask the gru: multi-task learning for deep text recommendations," in *Proceedings of the 10th ACM Conference on Recommender Systems*, pp. 107–114, ACM, 2016.

[19] J. Chen, W. Zhang, P. Zhang, P. Ying, K. Niu, and M. Zou, "Exploiting spatial and temporal for point of interest recommendation," *Complexity*, vol. 2018, Article ID 6928605, 16 pages, 2018.

[20] P. Zhao, X. Xu, Y. Liu, V. S. Sheng, K. Zheng, and H. Xiong, "Photo2trip: exploiting visual contents in geo-tagged photos for personalized tour recommendation," in *Proceedings of the 2017 ACM on Multimedia Conference - MM 17*, pp. 916–924, ACM Press, Mountain View, California, USA, October 2017.

[21] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for youtube recommendations," in *Proceedings of the 10th ACM Conference on Recommender Systems*, pp. 191–198, ACM, 2016.

[22] C.-Y. Wu, A. Ahmed, A. Beutel, A. J. Smola, and H. Jing, "Recurrent recommender networks," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 495–503, ACM, 2017.

[23] A. Beutel, P. Covington, S. Jain et al., "Latent cross: making use of context in recurrent recommender systems," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 46–54, ACM, Marina Del Rey, CA, USA, 2018.

[24] H. Yin, W. Wang, H. Wang, L. Chen, and X. Zhou, "Spatial-aware hierarchical collaborative deep learning for POI recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 11, pp. 2537–2551, 2017.

[25] H. Yin, X. Zhou, Y. Shao, H. Wang, and S. Sadiq, "Joint modeling of user check-in behaviors for point-of-interest recommendation," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 1631–1640, ACM, Melbourne, Australia, October 2015.

[26] P. Zhao, X. Xu, Y. Liu et al., "Exploiting hierarchical structures for POI recommendation," in *Proceedings of the 2017 IEEE International Conference on Data Mining (ICDM)*, IEEE, New Orleans, LA, USA, November 2017.

[27] P. Zhao, H. Zhu, Y. Liu et al., "Where to go next: a spatio-temporal gated network for next poi recommendation," in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI 2019)*, 2019.

[28] H. Wang, F. Zhang, X. Xie, and M. Guo, "Dkn: deep knowledge-aware network for news recommendation," in *Proceedings of the 2018 World Wide Web Conference*, pp. 1835–1844, Lyon, France, April 2018.

[29] Y. Gong and Q. Zhang, "Hashtag recommendation using attention-based convolutional neural network," in *Proceedings of the 25th International Joint Conference on Artificial Intelligence, IJCAI 2016*, pp. 2782–2788, NY, USA, July 2016.

[30] Y. Tay, A. T. Luu, and S. C. Hui, "Multi-pointer co-attention networks for recommendation," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2309–2318, London, UK, August 2018.

[31] Z. Cheng, Y. Ding, L. Zhu, and M. Kankanhalli, "Aspect-aware latent factor model: rating prediction with ratings and reviews," in *Proceedings of the 2018 World Wide Web Conference*, pp. 639–648, Lyon, France, April 2018.

[32] D. Tang, B. Qin, T. Liu, and Y. Yang, "User modeling with neural network for review rating prediction," in *Proceedings of the 24th International Conference on Artificial Intelligence, IJCAI 2015*, pp. 1340–1346, Argentina, July 2015.

[33] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[34] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical*

Methods in Natural Language Processing (EMNLP), pp. 1746–1751, Association for Computational Linguistics, Doha, Qatar, 2014, https://aclanthology.info/papers/D14-1181/d14-1181.

[35] Y. Zhang, Q. Ai, X. Chen, and W. B. Croft, "Joint representation learning for top-N recommendation with heterogeneous information sources," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 1449–1458, ACM, Singapore, Singapore, November 2017.

[36] J. Pennington, R. Socher, and C. Manning, "GloVe: global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

[37] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems*, Advances in neural information processing systems, pp. 3111–3119, Lake Tahoe, Nevada, 2013, https://dl.acm.org/citation.cfm?id=2999959.

[38] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," 2014, https://arxiv.org/abs/1412.6980.

[39] P. Gopalan, J. M. Hofman, and D. M. Blei, "Scalable recommendation with hierarchical Poisson factorization," in *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence, UAI 2015*, pp. 326–335, Netherlands, July 2015.

[40] X. Luo, M. Zhou, Y. Xia, and Q. Zhu, "An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1273–1284, 2014.

Hindawi
Complexity
Volume 2019, Article ID 8127869, 10 pages
https://doi.org/10.1155/2019/8127869

Research Article

Cognitive Driven Multilayer Self-Paced Learning with Misclassified Samples

Qi Zhu,[1,2] Ning Yuan,[1] and Donghai Guan[1]

[1]*College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China*
[2]*Corroborative Innovation Center of Novel Software Technology and Industrialization, Nanjing 210093, China*

Correspondence should be addressed to Donghai Guan; dhguan@nuaa.edu.cn

Received 29 November 2018; Accepted 12 February 2019; Published 10 March 2019

Guest Editor: Ke Deng

In recent years, self-paced learning (SPL) has attracted much attention due to its improvement to nonconvex optimization based machine learning algorithms. As a methodology introduced from human learning, SPL dynamically evaluates the learning difficulty of each sample and provides the weighted learning model against the negative effects from hard-learning samples. In this study, we proposed a cognitive driven SPL method, i.e., retrospective robust self-paced learning (R2SPL), which is inspired by the following two issues in human learning process: the misclassified samples are more impressive in upcoming learning, and the model of the follow-up learning process based on large number of samples can be used to reduce the risk of poor generalization in initial learning phase. We simultaneously estimated the degrees of learning-difficulty and misclassified in each step of SPL and proposed a framework to construct multilevel SPL for improving the robustness of the initial learning phase of SPL. The proposed method can be viewed as a multilayer model and the output of the previous layer can guide constructing robust initialization model of the next layer. The experimental results show that the R2SPL outperforms the conventional self-paced learning models in classification task.

1. Introduction

By assigning the samples in a meaningful learning order based on prior knowledge, curriculum learning (CL) [1] provides an easy-to-hard learning process, which makes the model more fits human cognition. To make curriculum learning more practical in dealing with machine learning problems, Kumar et al. [2] adaptively assessed the sample learning difficulty in model and proposed self-paced learning method. Specifically, self-paced algorithm actively and dynamically obtains the initial learning sequence from the original data and gradually increases the hard learning samples during each iteration. However, in curriculum learning, the sample learning course sequence is preset. By predefining or dynamically generating learning sequence, curriculum learning and self-paced learning can avoid main function falling into a bad local optimal solution. Many researchers applied curriculum learning and self-paced learning to some tough pattern recognition problems. In the literature [3], Jiang et al. proposed the self-paced curriculum learning, which not only obtains the dynamic sample sequence in the process of model learning, but also makes use of prior knowledge to avoid overfitting. Zhao et al. [4] applied the nonconvex problem of matrix decomposition, which suppresses effectiveness of the noise and outlier in the data on the model. Meanwhile, they pointed out that the strategy of adaptively selecting easy-learning sample sequences is similar to the process of human cognition. James et al. [5] adopted self-paced learning to SVM and achieved promising results in multimodal data retrieval. Self-paced learning has been introduced to many learning models and shown good performance in many real-world applications.

Self-paced learning seems to challenge the conventional learning methods, like active learning, boost, and transfer learning. In the view of machine learning, these boundary samples, noise samples, and outliers will increase the uncertainty of the model and may make the model generate a bad classification boundary. Therefore, compared to easy-learning samples, hard-learning learning samples have drawn much attention from the conventional model. In our work, we

aim to deal with supervised learning problems, in which easy-learning samples correspond to samples with small loss while hard-learning samples correspond to sample with large loss. In unsupervised learning, easy-learning samples mean the samples that are easy to be determined while hard-learning samples denote the samples that will cause the model to be unstable. In the paper, misclassified samples are denoted as the samples that the product of the predicted value and the label is negative. Typically, in AdaBoost learning, the model trains the classifier by changing the sample distribution based on the misclassified samples of previous iterations [6]. Li et al. [7] applied the sequence of AdaBoost to train classifiers, starting with weak learner and progressively boosted as a strong learner. Active learning is a kind of semisupervised learning, and it chooses to label the most valuable samples for the model. These low-confidence samples that may contain useful information are difficult to be chosen, which requires additional expert knowledge to identify. Tur et al. [8] presented a spoken language understanding method by combining active and semisupervised learning with human-label and automatically labeled data. Huang et al. [9] proposed a systematic framework to simultaneously measure the informativeness and the representativeness of an instance. The informativeness criteria reflects the ability of samples in reducing the uncertainty of model based on the labeled data, while the representativeness measures which samples can well represent the unlabeled data. However, self-paced learning model first considers the easy-learning samples with small prediction loss and gradually adopts hard-learning samples with larger prediction loss to extend the training set. The difference between self-paced learning and transfer learning is that the transfer learning improves the generalization of the model by sharing the models in different tasks [10], while the self-paced learning updates and learns itself to obtain the local optimal solution.

Study [11] pointed out the inherent consistency between human recognition and reinforcement learning. In dealing with a learning problem, humans and other animals utilize a harmonious combination of repeating learning and hierarchical sensory processing systems. In self-paced learning, the initial model is trained insufficiency with a few easy-learning samples, which increases the learning risk of follow-up iteration and even reduces the generalization of the final model. The usual practice of solving the small sample problem contains feature selection [12], regularization, adding artificial samples [13], etc. In order to improve the generalization of the initial model consisting of small samples in self-paced learning, we design the recurrent framework, which uses the model of last self-paced learning iteration to repeatedly construct the initial model. Corresponding to the repeating learning process of humans, if the initial model inherits the property of large sample learning model, the obtained final model may be more robust and discriminative.

Meanwhile, although self-paced learning and some conventional machine learning methods (AdaBoost, active learning and transfer learning) are very different in sample processing, we can still absorb the advantages of these conventional methods into self-paced learning. Specifically, in this paper, we propose retrospective robust self-paced learning (R2SPL). In each iteration of self-paced learning, besides considering easy-learning samples, these misclassified samples of last iteration will also be involved in training the model. For example, if the hard-learning samples (their categories are difficult to determine) in the data are the majority, conventional self-paced learning may not get a good local optimal solution. In this case, our proposed method focuses on both easy-learning samples and misclassified samples in each iteration, which can drive the final mature model be robust and discriminative.

Overall, our main contribution can be summarized as follows:

(i) We introduce these misclassified samples accompanied with easy samples with small loss in each iteration to guide the model becomes more discriminative.

(ii) Retrospective self-paced learning is proposed to improve the robustness of the initialization of self-paced learning.

(iii) Experiments results show the proposed method achieves promising result in classification tasks.

The remainder of this paper is organized as follows. We briefly introduce related works on self-paced learning in Section 2. We propose the robust SPL in Section 3. In Section 4, we conduct the experiments on UCI and ADNI datasets. We provide the conclusion and the future research plan in Section 5.

2. Related Work

2.1. Curriculum Learning and Self-Paced Learning. In 2009, Bengio et al. [1] proposed a method of imitating children education order which is called curriculum learning. Different from conventional machine learning methods obtained from overall sample learning, in their work, they sorted the samples in a meaningful order and learned the model in several sections. Benefiting from the prior knowledge, curriculum learning can get better results than other machine learning models in some tasks. However, arranging the sample order usually requires expert identification, which increases the difficulty and cost of the model. In addition, the ordered sample sequence is static and lacks flexibility in dealing with new samples or tasks. To alleviate this deficiency, Kumar et al. [2] proposed self-paced learning in 2010. Without any prior knowledge and expert identification, self-paced learning can dynamically assign the samples from easy to difficult based on the fitness between the samples and the model. In multimedia retrieval, Lu et al. [14] proposed self-paced reranking model for multimodal data, and the model made significant progress on both image and video search tasks. Zhou et al. [15] brought the self-paced learning to deep neural network, which can adaptively involve the faithful samples into training process. By analyzing the work mechanism of self-paced learning, Fan et al. [16] proposed a general implicit regularized framework. Since self-paced learning is adopted into many models, the commonality among these models lies in the sample processing. In each iteration,

these models usually pick these high-confidence samples which fit the model better to construct the current model and gradually use the remaining low-confidence samples to fine-tune the model to make it become more generalization.

Curriculum learning is the first attempt to combine human cognition sequence and machine learning model. Although curriculum learning has some drawbacks, it brings the idea of easy-to-hard learning to the latter models. Self-paced learning is the extension of curriculum learning, which is more flexible and concise. Similar to human learning, self-paced learning trains samples from easy to difficult and gradually improves the robustness of the model.

2.2. Tough Samples Learning. In the sample processing strategy, self-paced learning method is different with some tough samples focused learning methods, like AdaBoost, active learning, and transfer learning. In our work, we try to finely distinguish different types of samples, including easy-learning samples, hard-learning samples, and misclassified samples, and give them different weights in model. By combining the simple classifiers, AdaBoost can deal with complicated problem. For example, in many multiclass problems [17, 18], the distribution of samples is highly complex [19]. Like SVM, AdaBoost can asymptotically achieve a margin distribution which is robust to noise [7, 20, 21]. Active learning is a semisupervised model that uses the unlabeled samples to improve the model obtained by labeled samples. However, since the unlabeled samples have no tags, some data that are difficult to distinguish the types usually need to manually annotate. Otherwise, if these data are identified by the model, it may increase the uncertainty of the model. Lin et al. [22] proposed active self-paced learning that used the characteristic of these two models to automatically annotate the high-confidence and low-confidence samples and incorporated them into training under weak expert recertification. Kumar et al. [2] pointed out that certainty does not imply correctness. Many researchers performed SVM and active learning in some practical applications, like text classification [23, 24], image retrieval [25], and segmentation of images [26]. The model will adjust the weight of the data from original domain, which increases the similarity of the data between target domain and source domain [10]. In the process of children learning, some problems share a common underlying structure but differ in surface manifestations, which is similar to the characteristics of transfer learning [27]. In order to make the models close to human wisdom, many researchers combine the models with environmental feedback and transfer learning [28–30].

In our work, we will focus on both the easy-learning samples and the tough samples, which improve the discrimination of self-paced learning. In each iteration, we will simultaneously select these easy-learning samples and misclassified samples to train. Like human cognition, it is beneficial to improve the generalization of the final self-paced model by simultaneously learning the high-confidence samples and low-confidence samples in each iteration.

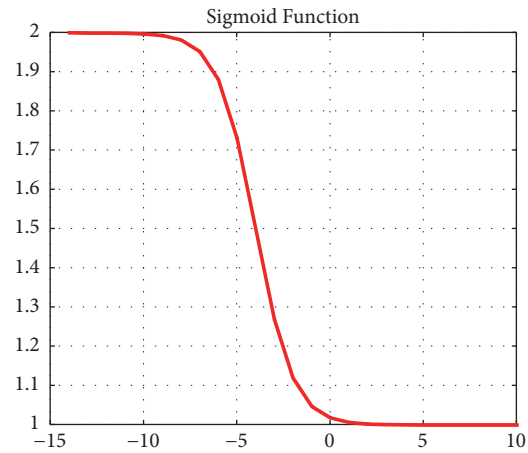


Figure 1: The weight function for weight matrix Q.

3. Proposed Method

3.1. Robust SPL. Specifically, we define a diagonal weight matrix $Q \in \mathbb{R}^{n\times n}$ to denote misclassified weight of each sample. Let y_i and $X_i^T\omega$ represent the label and predicted value of i-th sample, respectively. For binary classification problem, if $y_i X_i^T \omega^{(t)} > 0$, the i-th sample is corrected classified. Otherwise, this sample is considered as misclassified sample. In our work, the weight of these misclassified samples ($y_i X_i^T \omega^{(t)} < 0$) in weight matrix Q should be larger than these corrected samples, and the scope of this type of weight should not vary greatly. Therefore, in our work, we adopt sigmoid function, shown in Figure 1, as weight function with respect to the product of label and predicted value. Given the label vector $y \in \mathbb{R}^{1\times n}$, data matrix $X \in \mathbb{R}^{d\times n}$, and current model parameter $\omega^t \in \mathbb{R}^{d\times 1}$, the misclassified weight of i-th sample can be calculated as

$$q_{ii}^{(t+1)} = c_1 - \frac{1}{1 + \exp - \left(y_i X_i^T \omega^{(t)} + c_2\right)} \tag{1}$$

For supervised problem, self-paced learning function f(V; k) assigns weight to samples based on the sample loss. Those samples with small loss will be viewed as easy-learning samples. However, our model simultaneously considers easy-learning samples and tough samples in each iteration. Specifically, we combine Q and self-paced weight matrix V linearly. Then, the model can be formulated as

$$\begin{aligned} \text{as } L(\omega, V, Q) = \min_{\omega,V} \frac{1}{2} \left\| \left(y - \omega^T X\right)(V + Q)^{1/2} \right\|_2^2 \\ + \frac{\lambda}{2} \|\omega\|_p^p + f(V; k) \end{aligned} \tag{2}$$

where λ is the regularization term. To embed structure information in feature extraction, we adopt p-norm on the regression coefficient ω. The closer the value of p gets to 0, the sparser the result of the feature extraction is. f(V; k) is the self-paced weight function and k controls the number of samples which is considered to construct the model. At first, only a

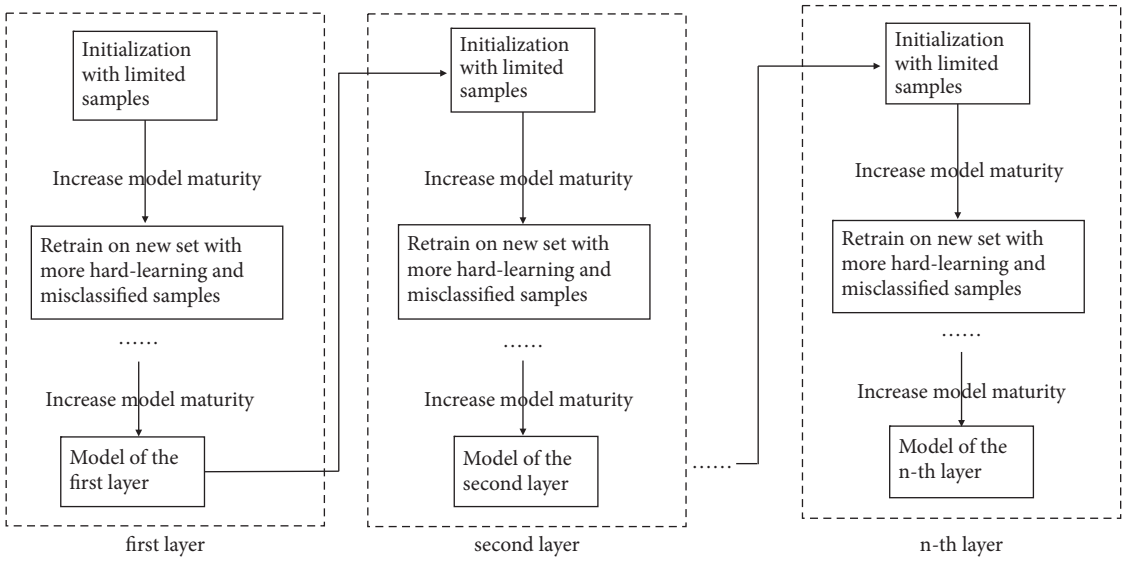


Figure 2: The network model of the proposed method.

few of samples with small loss can be utilized to construct the model. With the decrease of k, more and more samples with larger loss will join the model training process.

Whatever the forms of self-paced function $f(V; k)$ are, they should satisfy three properties [3, 14]: (1) $f(v; k)$ is convex with respect to $v \in [0, 1]$; (2) the sample weight should be monotonically decreasing with respect to its corresponding loss; (3) the sample weight should be monotonically decreasing with respect to the pace parameter k.

Meanwhile, in the process of human cognition, people usually make mistakes due to the lack of knowledge. By constantly summarizing unfamiliar and misunderstand concepts, people can form a more robust knowledge system. Notably, if the children get the help with adults (like teachers and parents) in the process of cognition, they can construct the knowledge framework more rapidly and soundly. The self-paced learning is similar to the education process of children without the help of adults. Therefore, the learned initial model may be not robust enough. To alleviate this deficiency, in this paper, we proposed retrospective robust self-paced learning. Specifically, we cascade multiple self-paced learning algorithms, which can help to reduce the negative impacts in the initialization of follow-up self-paced learning process due to lack of sample simple size problem. Naturally, in the next self-paced learning process, the learning rate can be speeded up moderately. Repeating the process for several times, we can obtain more robust and discriminative model. The framework of the proposed method can be viewed as a multilayer network shown in Figure 2. Firstly, we construct the initial model based on these easy-learning samples. The obtained initial model does not have good discriminability due to lack of sample training. Then, we adopt more hard-learning and misclassified samples to retrain our model, which drives our model to be more robust and discriminative. When the training of this layer is finished, the convergence results will be used as the prior knowledge of the next layer model. Repeat this operation until all n-layer models have been trained. Because self-paced learning stage is essentially a layer of the network. Specifically, the output of the first layer can be used to guide choosing of the easy-learning samples in the initialization of the second layer, which can be expected to be more robust than that learn independently.

3.2. Optimization. Since the parameters ω, Q, and V are independent with each other, we can fix other parameters when we calculate each of them. In t + 1-th iteration, each parameter can be calculated by the t-th iteration parameters.

$$\begin{aligned} \omega^{t+1} &= \min_{\omega} L\left(\omega, V^t, Q^t\right) \\ V^{t+1} &= \min_{V} L\left(\omega^t, V, Q^t\right) \\ Q^{t+1} &= \min_{Q} L\left(\omega^t, V^t, Q\right) \end{aligned} \tag{3}$$

In t + 1-th iteration, each portion is a convex problem; the optimal solutions of parameters can be achieved. The solutions of ω, Q, and V are presented as follows.

(1) The Solution of ω

$$\begin{aligned} L(\omega, V, Q) = \min_{\omega, V} &\frac{1}{2} \left\| \left(y - \omega^T X\right) (V + Q)^{1/2} \right\|_2^2 \\ &+ \frac{\lambda}{2} \|\omega\|_p^p \end{aligned} \tag{4}$$

To simplify the calculation, we convert the second term of (4) to $\lambda/2\mathrm{tr}(\omega^{\mathrm{T}} G \omega)$. $G \in \mathbb{R}^{d \times d}$ is a diagonal matrix and the diagonal elements can be calculated by

$$g_{ii} = \omega_i^{p-2} \tag{5}$$

1: Input:
2: X_{train}: source domain data;
3: X_{test}: target domain data;
4: k, k′: self-paced parameter; λ: regularization parameter
5: For each layer of self-paced learning:
6: Initiate parameter ω by utilizing the result of last self-paced learning layer.
7: Repeating until convergence
8: Update V by Eq. (12);
9: Update Q by Eq. (1);
10: Update ω by Eq. (8);
11: Train the model based on V and Q.

Algorithm 1: The algorithm of robust self-paced learning (R2SPL).

where ω_i is the i-th element of ω. Then, (4) can be equivalently formulated as

$$\min_{\omega} \frac{1}{2} tr\left(\left(y - \omega^T X\right)(V+Q)\left(y - \omega^T X\right)^T\right) + \frac{\lambda}{2} tr\left(\omega^T G \omega\right) \tag{6}$$

Get the derivation of ω in (6) and set it to 0:

$$X(V+Q)\left(X^T\omega - y^T\right) + \lambda G\omega = 0 \tag{7}$$

Then, the optimal solution of ω is

$$\omega = \left(X(V+Q)X^T + \lambda G\right)^{-1}\left(X(V+Q)y^T\right) \tag{8}$$

In the next iteration, the model will correct its mistake by guiding the regression coefficient ω based on the parameter Q. Under the influence of the accumulation of V and Q, which corresponds to easy samples with small loss and misdirected samples, our proposed self-paced learning model will be more robust and discriminative than conventional self-paced learning models which only consider easy sample in each iteration.

(2) The Solution of V

$$L(\omega, V, Q) = \min_{V} \frac{1}{2}\left\|\left(y - \omega^T X\right)(V+Q)^{1/2}\right\|_2^2 + f(V;k) \tag{9}$$

In our work, we define f(V; k) as

$$f(V;k) = -\zeta \sum_{i=1}^{n} \log\left(v_{ii} + \zeta k\right) \tag{10}$$

where $\zeta = 1/(k' - k)$, $1/k'$ is used to describe the lower bound of sample loss, and $1/k$ is used to describe the upper bound. Meanwhile, $1/k$ also describes the age of the model. In the initial stage, only easy samples with small loss are considered to construct the model. As $1/k$ grows, more and more complicated samples with larger loss will be adopted to the model to make it more mature. The sample weight can be calculated by our self-paced weight function f(V; k). Get the derivation of V in (9) and set it to 0:

$$\ell_i - \frac{\zeta}{v_{ii} + \zeta k} = 0 \tag{11}$$

where ℓ_i is the squared loss of i-th sample. Then, the optimal solution of V is given by

$$v_{ii} = \begin{cases} 1 & \ell_i \le \dfrac{1}{k'} \\ 0 & \ell_i \ge \dfrac{1}{k} \\ \dfrac{\zeta}{\ell_i} - k\zeta & otherwise \end{cases} \tag{12}$$

As mentioned above, we adopt retrospective self-paced learning framework to increase the robustness and discrimination of model. Specifically, the step size of k in the current self-paced learning process is smaller than that in the follow-up process. In our work, we set the step size of first self-paced learning layer is 0.1 and gradually increase it in follow-up process. To simplify the calculation, we set the number of layers to 3 in our method.

(3) The Solution of Q. The solution of Q can be calculated by (1). In our work, we apply sigmoid function to assign weight value to matrix Q. Using different self-paced weight functions in (10), we can obtain different models. In detail, we adopt three self-paced learning function, binary, linear, and logarithmic. f(V; k) can be formulated as follows.

(a) Binary

$$f(V;k) = -\frac{1}{k}\|V\|_1 \tag{13}$$

(b) Linear

$$f(V;k) = \frac{1}{k}\left(\frac{1}{2}\|V\|_2^2 - \sum_{i=1}^{n} v_{ii}\right) \tag{14}$$

(c) Logarithmic

$$f(V;k) = \sum_{i=1}^{n}\left(\xi v_{ii} - \frac{\xi v_{ii}}{\log \xi}\right) \tag{15}$$

where $\zeta = (k-1)/k$. The solving algorithm of our model is shown in Algorithm 1.

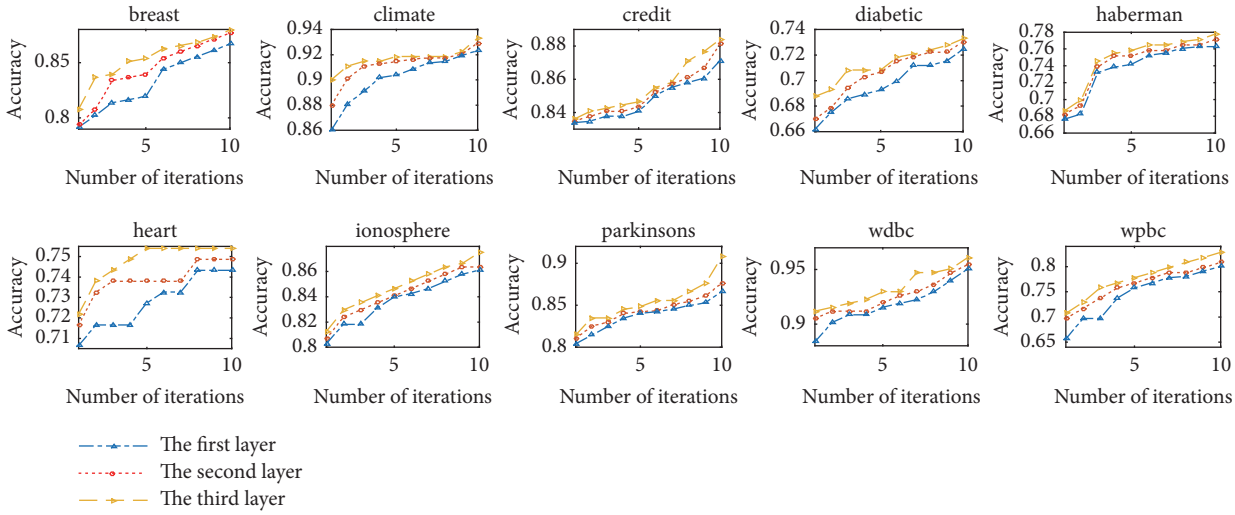


Figure 3: The classification results of the three models on the top ten iterations.

Table 1: Notations of UCI databases.

Binary-class Dataset		
Dataset	Dimension	Number
wpbc	33	151/47
wdbc	30	212/357
parkinsons	22	147/48
ionosphere	225	126/99
heart	22	157/110
haberman	3	225/81
diabetic	19	540/611
credit	15	307/383
climate	18	46/494
breast	9	458/241

4. Experiments

4.1. Settings. To evaluate the effectiveness of our proposed method, we conduct our experiment on ten binary classification datasets from UCI repository and Alzheimer's Disease Neuroimaging Initiative (ADNI) database. The detailed information of UCI datasets is presented in Table 1. AD data used in our experiment is obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database (adni.loni.usc.edu). In our work, the Alzheimer's Disease (AD) data have 913 samples with 116-dimension, which are consisting of 160 AD patients, 542 MCI patients, and 211 healthy controls (HC). Specifically, the MCI patients can be divided into three stages, 82 Significant Memory Concern (SMC) patients, 273 Early Mild Cognitive Impairment (EMCI) patients, and 187 Late Mild Cognitive Impairment (LMCI) patients. There are five modalities in the ADNI data, including ID (serial number), single nucleotide polymorphism (SNPdata), voxel based morphometry (VBM), fluorodeoxyglucose position emission tomography (FDG), and F-18 florbetapir PET scans amyloid imaging (AV45). In ADNI database, we perform three classification tasks, AD versus HC, MCI versus HC, and SMC versus LMCI. In each classification task, we compare our method with baselines SVM with RBF kernel, AdaBoost and conventional self-paced methods. In the sample processing, AdaBoost adjusts the distribution of training samples based on the performance of basic learners, which makes the misdirected samples in current iteration get more attention in the next iteration.

For conventional self-paced learning models whose weight functions are (13), (14), and (15), we define them as binary, linear, and log for short. Meanwhile, we construct two self-paced learning models based on our proposed models. If the parameter Q is not considered into the retrospective model, we call the model as Easy-SPL for short. When our proposed model is just one level self-pace learning containing parameter Q, it can be defined as Single-SPL. To obtain unbiased results, we adopt 10-fold cross-validation strategy with four measurements, including classification accuracy (ACC), sensitivity (SEN), specificity (SPE), and area under receiver operating characteristic curve (AUC). In UCI databases, we repeat all experiments 30 times with 2-folds cross-validation. In the experiment, we analyze the results of each layer of self-paced leaning process to determine the number of layers. We stop training the model when the convergence results of current self-paced learning process are not significantly improved compared with the previous iteration process. Then, we can determine the number of layers in our model.

4.2. Experimental Results on UCI and ADNI Data. At first, we verify the effectiveness of introducing tough samples and retrospective self-paced learning to the model in each iteration on ten UCI datasets. Figure 3 lists the results of each layer of the proposed self-paced learning method. Obviously, after introducing weight matrix Q, the model behaves more discriminative in each iteration. Meanwhile, the model of last self-paced learning process not only has better performance but also behaves more robust than the previous layer. Table 2

Table 2: Results of seven baselines and our model on 10 UCI datasets.

Dataset		SVM	AdaBoost	Binary	Linear	Log	Easy-SPL	Single-SPL	R2SPL
wpbc	ACC	75.08±0.021	76.04±0.023	76.07±0.008	76.33±0.025	76.20±0.017	76.33±0.009	72.86±0.019	77.58±0.004
	AUC	67.10±0.054	73.18±0.035	74.32±0.063	74.22±0.013	71.11±0.031	50.85±0.019	61.00±0.028	74.47±0.036
wdbc	ACC	92.84±0.008	93.67±0.021	92.88±0.018	93.64±0.071	93.71±0.044	91.34±0.029	94.08±0.018	94.57±0.006
	AUC	98.20±0.052	98.62±0.021	98.19±0.022	98.33±0.031	98.38±0.007	96.79±0.066	93.59±0.034	98.68±0.055
parkinsons	ACC	84.67±0.028	85.22±0.038	85.57±0.037	84.23±0.011	84.16±0.031	77.49±0.041	82.61±0.058	85.74±0.022
	AUC	83.13±0.031	86.22±0.014	84.11±0.053	81.89±0.009	81.26±0.054	83.85±0.071	76.47±0.019	87.43±0.055
ionosphere	ACC	85.42±0.029	85.97±0.017	85.40±0.034	84.43±0.061	85.23±0.019	76.65±0.008	87.29±0.011	88.37±0.012
	AUC	96.65±0.057	97.14±0.047	96.48±0.061	96.91±0.031	96.77±0.058	88.71±0.019	83.39±0.022	97.23±0.046
heart	ACC	81.82±0.033	82.35±0.031	83.96±0.005	86.63±0.004	87.70±0.061	75.40±0.019	64.17±0.021	89.84±0.023
	AUC	81.09±0.049	80.43±0.012	79.84±0.009	82.36±0.017	83.02±0.032	84.22±0.045	50.23±0.028	85.54±0.046
haberman	ACC	73.07±0.062	73.55±0.019	74.20±0.028	73.51±0.066	73.66±0.057	72.79±0.015	72.24±0.025	74.42±0.022
	AUC	63.54±0.011	67.32±0.015	65.91±0.023	68.33±0.001	68.48±0.068	61.47±0.039	58.62±0.027	68.52±0.003
diabetic	ACC	68.60±0.029	67.59±0.037	67.80±0.011	69.49±0.016	68.76±0.055	54.54±0.044	63.38±0.021	69.91±0.017
	AUC	76.53±0.022	76.39±0.015	75.94±0.008	76.92±0.064	76.37±0.016	52.50±0.049	63.77±0.054	77.34±0.015
credit	ACC	85.82±0.016	85.59±0.068	85.35±0.033	85.63±0.005	85.53±0.051	83.09±0.062	85.18±0.011	85.89±0.033
	AUC	88.74±0.029	89.32±0.014	89.46±0.009	89.44±0.035	88.62±0.026	89.03±0.003	85.10±0.049	89.53±0.061
climate	ACC	91.41±0.033	91.20±0.059	91.48±0.064	91.25±0.029	91.41±0.019	91.06±0.006	92.33±0.015	91.53±0.055
	AUC	86.79±0.016	89.48±0.023	83.39±0.055	89.75±0.036	87.72±0.018	89.37±0.049	65.68±0.048	90.41±0.017
breast	ACC	96.91±0.039	96.82±0.064	96.41±0.033	96.62±0.019	96.45±0.026	92.28±0.016	94.50±0.017	97.13±0.014
	AUC	99.40±0.023	99.43±0.039	99.28±0.008	99.28±0.046	99.29±0.022	98.66±0.033	93.63±0.028	99.47±0.011

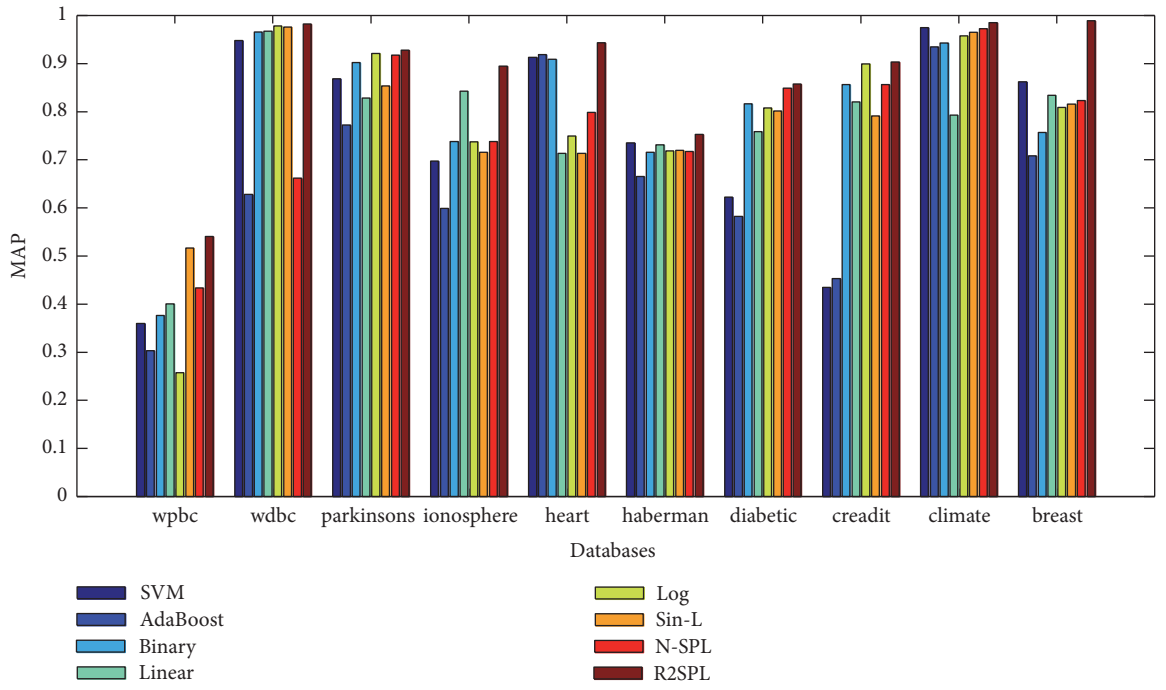


Figure 4: The classification results of state-of-art methods and our model on ten databases.

lists the ACC and AUC of seven baselines and our model. Our proposed model achieves all the best results on 10 UCI datasets and makes great improvement in several datasets.

We compared the proposed method, i.e., R2SPL, with several representative classification methods, including SVM, AdaBoost, SPL with binary, linear, or log function, Single-SPL (Sin-L), SPL without tough samples (N-SPL). The results are shown in Figure 4. We draw the precision-recall curves of these methods in Figure 5 and presented AUC and ACC results in Table 2. As seen from Figures 4 and 5 and Table 2, we find our methods outperform these comparison methods on all the ten datasets.

We also performed our method and comparison methods on ADNI dataset and conducted three classification tasks,

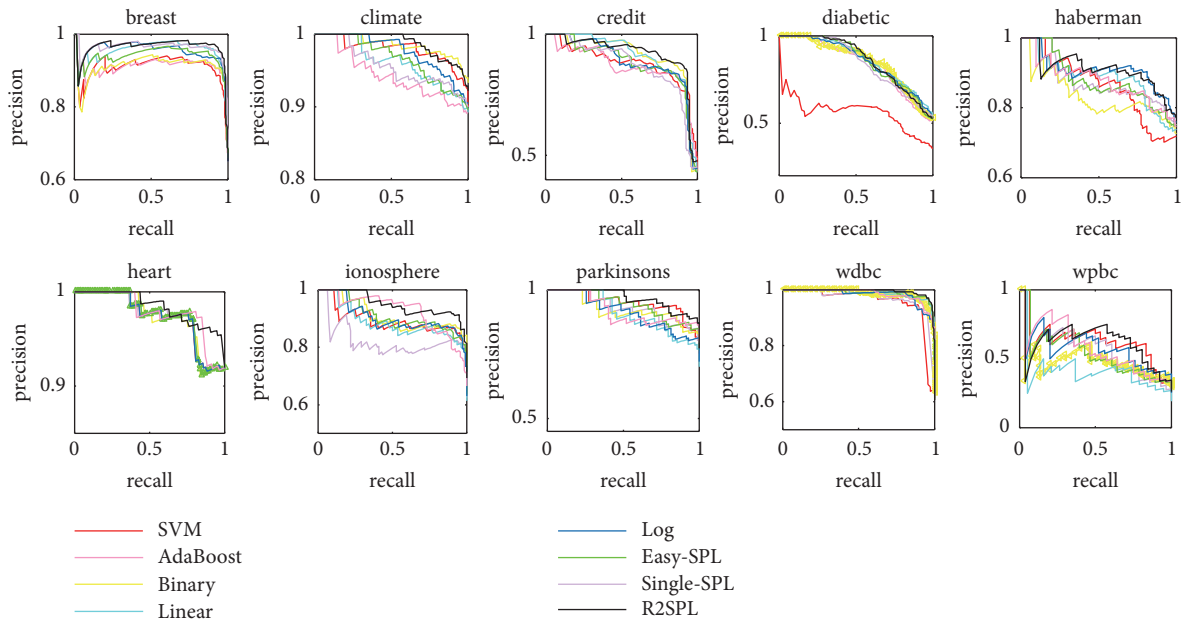


Figure 5: The precision-recall (PR) figures of state-of-the-art methods and our model on 10 UCI databases.

Table 3: Results of seven baselines and our model on the task of AD versus HC.

	SVM	Ada-Boost	Binary	Linear	Log	N-SPL	Single-SPL	R2SPL
ACC	78.31±0.034	87.45±0.021	74.12±0.019	71.80±0.028	71.29±0.056	89.57±0.033	90.09±0.057	91.34±0.071
SEN	89.09±0.028	90.83±0.047	75.56±0.035	66.38±0.044	79.79±0.068	90.57±0.027	90.46±0.026	92.38±0.059
SPE	62.21±0.019	82.90±0.051	73.88±0.080	80.32±0.037	60.94±0.038	89.64±0.068	90.62±0.087	90.99±0.096
AUC	80.70±0.022	83.74±0.034	74.65±0.029	72.78±0.011	70.12±0.042	90.35±0.019	91.68±0.027	92.62±0.028

Table 4: Results of seven baselines and our model on the task of MCI versus HC.

	SVM	Ada-Boost	Binary	Linear	Log	N-SPL	Single-SPL	R2SPL
ACC	66.61±0.021	83.68±0.028	71.77±0.059	73.52±0.027	73.01±0.019	80.32±0.097	80.12±0.075	84.09±0.069
SEN	50.24±0.035	80.24±0.034	48.27±0.022	62.12±0.038	56.84±0.057	70.13±0.032	71.62±0.062	73.42±0.095
SPE	82.99±0.068	81.56±0.019	76.35±0.021	84.91±0.044	89.18±0.051	90.52±0.058	88.61±0.078	94.76±0.063
AUC	67.16±0.026	82.49±0.038	57.40±0.051	70.68±0.035	71.21±0.080	78.96±0.049	82.58±0.072	86.49±0.082

Table 5: Results of seven baselines and our model on the task of SMC versus LMCI.

	SVM	Ada-Boost	Binary	Linear	Log	N-SPL	Single-SPL	R2SPL
ACC	69.50±0.043	66.89±0.042	73.63±0.028	72.08±0.053	75.07±0.034	69.53±0.053	69.86±0.066	75.81±0.033
SEN	46.75±0.029	48.63±0.036	54.70±0.055	59.46±0.032	65.16±0.056	48.00±0.035	60.94±0.087	48.32±0.016
SPE	70.13±0.093	73.59±0.028	74.18±0.090	69.03±0.046	70.71±0.042	72.87±0.024	66.48±0.055	80.72±0.097
AUC	71.40±0.045	64.74±0.014	74.79±0.038	74.53±0.031	79.40±0.089	64.93±0.048	69.48±0.019	72.93±0.029

AD versus HC, MCI versus HC, and SMC versus LMCI. The comparison results in three tasks, AD versus HC, MCI versus HC, and SMC versus LMCI, are listed in Tables 3, 4, and 5, respectively. Obviously, our proposed method has better performance compared with other methods in ACC, SEN, SPE, and AUC. It demonstrates the superiority of our model to other classifiers in AD classification problems.

4.3. Parameter Influence. Our model has two parameters including regularization term λ and sparse term p. We test the influence of the two parameters on 10 UCI datasets. The parameter λ is tuned from 10^{-3} to 10^{3} and the value of sparse term p is adjusted from 0 to 2. When detecting the sensitivity of a parameter to the model, we only change the value of this parameter and fix the value of another parameter. Figure 6 shows the experimental results. Specifically, Figure 6 shows the influence of regularization term λ and parameter p. As we can see from Figure 6, when the λ is tuned from 10^{-3} to 10^{3}, the performance of our proposed model is stable in most cases. Figure 6 also shows the influence

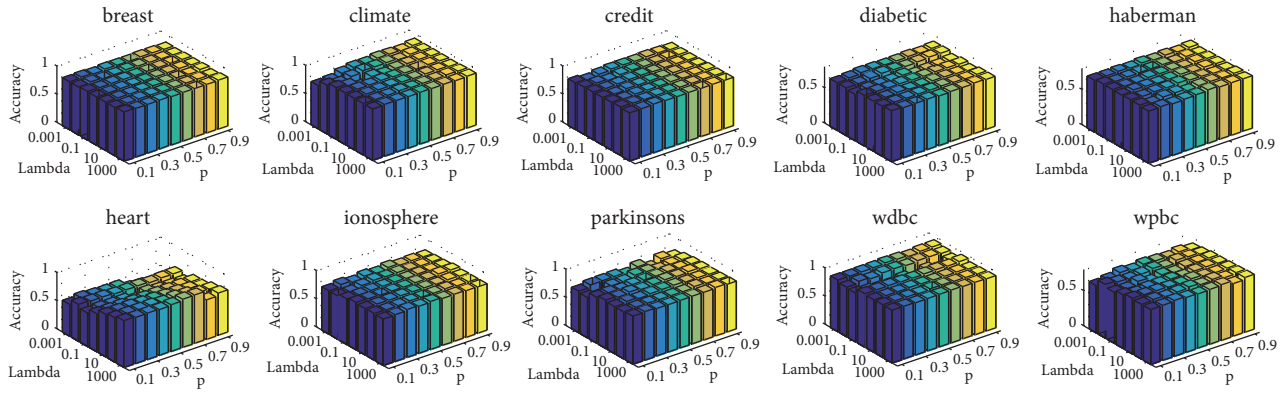


Figure 6: The results of our model base different parameters.

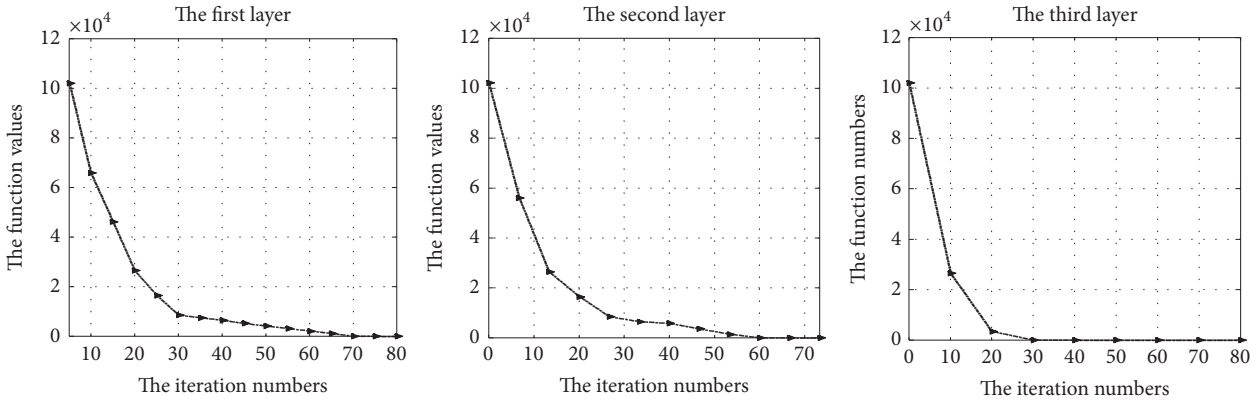


Figure 7: The convergence results of three layers models.

of sparse term p. When p changes from 0.4 to 2, the performance of our proposed method changes slightly. We conduct multiple groups of experiment on 10 UCI datasets. The experiment results verify that our model is not sensitive to specific parameters and only related to the structure of the model.

4.4. The Convergence Results of Different Layers of Model. In the convergence analysis, we find that different models have different rates of convergence. The convergence results are listed in Figure 7; obviously, as the number of layers increases, the convergence speed of the model is also accelerating. Benefiting from the prior knowledge of previous iteration process, the current model can obtain local optimal solution faster.

5. Conclusion

In this paper, we divide the samples into easy-learning samples, hard learning samples, and misclassified samples and analyze their roles in learning. Then, we introduce tough or misclassified sample in the training of each iteration to self-paced learning. Meanwhile, considering the human cognition process, people usually need to constantly explore and learn from the same data or task to obtain a deep knowledge about it by multiple learning stages. So, we design the retrospective framework to improve the robust of self-paced learning, which uses the model in previous layer to reduce the negative effect of small sample size problem in the initialization phase of next iteration. The experimental results show that the proposed method behaves more robust and discriminative than conventional self-paced learning methods and many representative methods. In our further work, we will extend above framework to other learning tasks, such as semisupervised learning.

Data Availability

Raw data were generated at Nanjing University of Aeronautics and Astronautics. Derived data supporting the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China (nos. 61501230, 61732006,

61876082, and 61861130366), National Science and Technology Major Project (no. 2018ZX10201002), and the Fundamental Research Funds for the Central Universities (no. NP2018104).

References

[1] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th International Conference On Machine Learning, ICML 2009*, pp. 41–48, Canada, June 2009.

[2] M. P. Kumar, B. Packer, and D. Koller, "Self-paced learning for latent variable models," *Advances in Neural Information Processing Systems*, pp. 1189–1197, 2010.

[3] L. Jiang, D. Meng, Q. Zhao, S. Shan, and A. G. Hauptmann, "Self-paced curriculum learning," in *Proceedings of the AAAI*, vol. 2, p. 6, 2015.

[4] Q. Zhao, D. Meng, L. Jiang, Q. Xie, Z. Xu, and A. G. Hauptmann, "Self-paced learning for Matrix factorization," in *Proceedings of the 29th AAAI Conference on Artificial Intelligence, AAAI 2015 and the 27th Innovative Applications of Artificial Intelligence Conference, IAAI 2015*, pp. 3196–3202, 2015.

[5] J. S. Supancic III and D. Ramanan, "Self-paced learning for long-term tracking," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2013*, pp. 2379–2386, June 2013.

[6] C. Ying, M. Qi-Guang, L. Jia-Chen, and G. Lin, "Advance and prospects of AdaBoost algorithm," *Acta Automatica Sinica*, vol. 39, no. 6, pp. 745–758, 2013.

[7] X. Li, L. Wang, and E. Sung, "AdaBoost with SVM-based component classifiers," *Engineering Applications of Artificial Intelligence*, vol. 21, no. 5, pp. 785–795, 2008.

[8] G. Tur, D. Hakkani-Tür, and R. E. Schapire, "Combining active and semi-supervised learning for spoken language understanding," *Speech Communication*, vol. 45, no. 2, pp. 171–186, 2005.

[9] S. Huang, R. Jin, and Z.-H. Zhou, "Active learning by querying informative and representative examples," *Advances in neural information processing systems*, pp. 892–900, 2010.

[10] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[11] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[12] Z. Zhang, Y. Xu, J. Yang, X. Li, and D. Zhang, "A survey of sparse representation: algorithms and applications," *IEEE Access*, vol. 3, pp. 490–530, 2015.

[13] D.-C. Li, C.-S. Wu, T.-I. Tsai, and Y.-S. Lina, "Using mega-trend-diffusion and artificial samples in small data set learning for early flexible manufacturing system scheduling knowledge," *Computers & Operations Research*, vol. 34, no. 4, pp. 966–982, 2007.

[14] L. Jiang, D. Meng, T. Mitamura, and A. G. Hauptmann, "Easy samples first: Self-paced reranking for zero-example multimedia search," in *Proceedings of the 2014 ACM Conference on Multimedia, MM 2014*, pp. 547–556, USA, November 2014.

[15] S. Zhou, J. Wang, D. Meng et al., "Deep self-paced learning for person re-identification," *Pattern Recognition*, vol. 76, pp. 739–751, 2018.

[16] Y. Fan, R. He, J. Liang, and B.-G. Hu, "Self-paced learning: an implicit regularization perspective," in *Proceedings of the AAAI*, vol. 3, p. 4, 2017.

[17] T. Hastie, S. Rosset, J. Zhu, and H. Zou, "Multi-class adaboost," *Statistics and Its Interface*, vol. 2, no. 3, pp. 349–360, 2009.

[18] F. Lv and R. Nevatia, "Recognition and segmentation of 3-d human action using hmm and multi-class adaboost," in *Proceedings of the European Conference on Computer Vision*, pp. 359–372, Springer, 2006.

[19] P. Viola and M. Jones, "Fast and robust classification using asymmetric adaboost and a detector cascade," *Advances in Neural Information Processing Systems*, pp. 1311–1318, 2002.

[20] G. Rätsch, T. Onoda, and K. R. Müller, "Soft margins for AdaBoost," *Machine Learning*, vol. 42, no. 3, pp. 287–320, 2001.

[21] J. H. Morra, Z. Tu, L. G. Apostolova, A. E. Green, A. W. Toga, and P. M. Thompson, "Comparison of adaboost and support vector machines for detecting alzheimer's disease through automated hippocampal segmentation," *IEEE Transactions on Medical Imaging*, vol. 29, no. 1, pp. 30–43, 2010.

[22] L. Lin, K. Wang, D. Meng, W. Zuo, and L. Zhang, "Active Self-Paced Learning for Cost-Effective and Progressive Face Identification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 1, pp. 7–19, 2018.

[23] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *Journal of Machine Learning Research*, vol. 2, pp. 45–66, 2001.

[24] G. Schohn and D. Cohn, "Less is more: Active learning with support vector machines," in *Proceedings of the ICML*, pp. 839–846, Citeseer, 2000.

[25] S. Tong and E. Chang, "Support vector machine active learning for image retrieval," in *Proceedings of the 9th ACM International Conference on Multimedia*, pp. 107–118, October 2001.

[26] P. Mitra, B. U. Shankar, and S. K. Pal, "Segmentation of multispectral remote sensing images using active support vector machines," *Pattern Recognition Letters*, vol. 25, no. 9, pp. 1067–1074, 2004.

[27] A. L. Brown and M. J. Kane, "Preschool children can learn to transfer: Learning to learn and learning from example," *Cognitive Psychology*, vol. 20, no. 4, pp. 493–523, 1988.

[28] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: a survey," *Journal of Machine Learning Research*, vol. 10, pp. 1633–1685, 2009.

[29] H. Shin, H. R. Roth, M. Gao et al., "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning," *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.

[30] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015*, pp. 3156–3164, June 2015.

Hindawi
Complexity
Volume 2019, Article ID 4027638, 10 pages
https://doi.org/10.1155/2019/4027638

Research Article

Edge Computing in an IoT Base Station System: Reprogramming and Real-Time Tasks

Huifeng Wu,[1] **Junjie Hu**,[2] **Jiexiang Sun**,[3] **and Danfeng Sun**[4]

[1]*Institute of Intelligent and Software Technology, Hangzhou Dianzi University, Hangzhou 310018, China*
[2]*Hangzhou Yiyitaidi Information Technology Co., Ltd., Hangzhou 310000, China*
[3]*Beijing Research Institute of Automation for Machinery Industry Co., Ltd., Beijing 100120, China*
[4]*Institut f. Automation und Kommunikation, Magdeburg 39106, Germany*

Correspondence should be addressed to Danfeng Sun; danfeng.sun@ifak.eu

Received 14 December 2018; Accepted 10 February 2019; Published 5 March 2019

Guest Editor: Jiajie Xu

There are millions of base stations distributed across China, each containing many support devices and monitoring sensors. Conventional base station management systems tend to be hosted in the cloud, but cloud-based systems are difficult to reprogram and performing tasks in real-time is sometimes problematic, for example, sounding a combination of alarms or executing linked tasks. To overcome these drawbacks, we propose a hybrid edge-cloud IoT base station system, called BSIS. This paper includes a theoretical mathematical model that demonstrates the dynamic characteristics of BSIS along with a formulation for implementing BSIS in practice. Embedded programmable logic controllers serve as the edge nodes; a dynamic programming method creates a seamless integration between the edge nodes and the cloud. The paper concludes with a series of comprehensive analyses on scalability, responsiveness, and reliability. These analyses indicate a possible 60% reduction in the number of alarms, an edge response time of less than 0.1s, and an average downtime ratio of 0.66%.

1. Introduction

A base station is an information exchange center for the smartphones within its coverage area. These networks of base stations are the backbone of a mobile network and, for many, that means the backbone of one's work, life, or social sphere. A defect in any one of those base stations can mean great inconvenience for thousands of users. Typically, a base station consists of numerous devices that cooperate to ensure the base station's reliability, while countless sensors linked to those devices constantly assess the surrounding environment. If even one parameter value exceeds its threshold, alarms begin to sound. The incident is uploaded to the cloud server as a maintenance request and, once the validity of the alarm is confirmed, maintenance personnel are called to respond to the request.

Advances in cloud-based computing and storage have contributed to the thriving success of centralized systems. Yet, no matter how advanced, the reality of managing millions of base stations and monitoring hundreds of parameters for each and every one brings some stark practical problems to the fore.

(1) It is beyond the ability of any maintenance team to respond to every alarm or even respond to all priority alarms in time. In fact, many alarms are simply ignored.

(2) Due to the low response times inherent to cloud-based platforms, some urgent and real-time tasks must be managed by edge nodes. Executing an action at the instant an alarm sounds is one such example.

(3) The edge nodes and the central platform are not integrated seamlessly. Further, even though edge-node software can be updated remotely in some platforms, such updates can only occur en masse, and the fundamental program can only be updated not replaced entirely. However, the need to reprogram the tasks an edge node can perform is increasing, and every edge node may not necessarily need to perform

the same task. Combined alarms that are based on just a few relevant parameters are a good example, as these signals could be monitored by a small proportion of the network's nodes.

To address these problems, we propose a new architecture for base station management that combines edge and cloud computing, called BISI, which is presented in this paper. BISI

(1) reduces the incidence of unnecessary alarms and improves overall system reliability

(2) strengthens edge node capability to form a seamless edge-cloud system

(3) increases the scalability of the system and the dynamism of the edge node's functions

Edge computing has the advantages of reduced latency, less and lower traffic peaks, and longer node lifetimes [1, 2]. Hence, with an edge-cloud computing system, more functions can be shifted from the cloud to the edge nodes, especially real-time tasks, and those nodes can be reprogrammed dynamically. Moreover, the interactions between the cloud and the edge can be optimized.

Hence, the contributions of our study are summarized below.

(1) This paper presents a theoretical mathematical model with dynamic characteristics for an IoT base station management system. Abstract definitions are provided for the cloud and edge components, the interactions between the cloud and the edge nodes, and the interactions between the edge nodes and the devices.

(2) We explain how BSIS might be implemented in practice. In this paper, we use embedded programmable logic controllers (ePLCs) as the edge nodes with a three-layer software structure and a local database. The ePLCs comply with IEC 61131-3 standards [3]. With support from the cloud, the edge nodes can be programmed dynamically, can be reprogrammed, and can perform real-time tasks, such as linked tasks, ringing a combination of alarms, or filtering alarms. Therefore, the edge nodes are scalable and different edge nodes can run different numbers and types of tasks.

(3) Comprehensive analyses of BISI verify its scalability, response times, and reliability and demonstrate the advances an edge-cloud system can make in base station management.

The remainder of this paper is structured as follows: Section 2 introduces the related works; Section 3 presents the architecture of the BSIS; Section 4 provides the scalability, response time, and reliability analyses; and the last section concludes our work.

2. Related Works

With IoT platforms, good application performance is heavily dependent on the system architecture. Given the millions of edge nodes in base station networks, appropriate system architecture must be chosen carefully and thoughtfully. In our context, this consideration is even more critical if BSIS is to deliver a responsive, reliable, and scalable system, as the related studies on three key aspects of BSIS illustrate below.

2.1. System Structure. Cloud platforms have become one of the mainstays in computing due to their ability to deliver elastic computing power and storage to satisfy the needs of resource-constrained end-user devices [4]. However, edge computing has its own advantages, such as reducing latency, avoiding traffic peaks, and extending the life of an edge node [1, 2]. Hence, combining edge computing with a cloud platform was inevitable and has given rise to a new paradigm [5]. Ever since researchers have sought ways to efficiently distribute computationally intensive tasks to leverage the respective advantages of each, many studies have proposed a resource management approach that focuses on a particular function, e.g., admission control, computational resource allocation, and power control [6–8]. Fu et al. [9] illustrated secure data storage and searching method to improve the efficiency and security of data storage and retrieval. Chekired et al. [10] proposed a two-priority queuing model to schedule and analyze industrial data. Suganuma et al. [11] proposed a flexible and advanced system model.

Moreover, edge-based cloud platforms have found applications in a variety of domains. For example, Jia et al. [12] introduced a manhole cover management system, which boasts good response times. Xu et al. [13] outlined a smart grid system to effectively integrate energy resources with storage devices. Pace et al.'s research [14] focuses on emerging healthcare industries. Smart transportation systems are another hot domain [15, 16].

However, applying the various solutions found in the studies mentioned above to base station management is problematic as few consider the complexity associated with scalability, reliability, and responsiveness.

Some software approaches to implementing IoT systems have also been developed, for example, multiagent systems and service-oriented architecture (SOA). In multiagent systems, every function or entity can be described as an agent and tasks are completed in the cloud through positive or negative interactions between these agents [17–19]. This approach provides a method with which to clearly describe complex systems that contain many functions and entities linked through complex relationships. Perhaps advantageously, BSIS cannot benefit from a multiagent approach because its functions and the types of entities are both limited and unambiguous.

SOA is a discrete functionality that relies on standardized interfaces to form applications. The SOA paradigm offers loose-coupling, reconfigurability, and flexibility, which allows service entities to be composed or orchestrated during runtime to create different system compositions [20]. Thus, the two main characteristics of SOA are autonomy and interoperability [21, 22]. BSIS shares one similar concept, i.e., scalability. However, SOAs cannot generally handle sensitive real-time tasks [21], and the scalability of an SOA is often limited to its upper layers.

2.2. Edge Node. An edge node is any computing or network resource on the path between a data source and the center of the cloud [5]. In the body of research on edge computing, many devices have served as edge nodes, e.g., smartphones [23], routers [24].

PLCs are common in industrial fields [25–29] because they are highly reliable and easy to program [30]. Moreover, they were standardized as early as the 1990s under IEC61131-3 [3]. The PLCopen organization continues the drive for standardization to this day in areas such as motion control [31], TC4 for integration, OPC UA and TC5 for safety, and TC6 for seamless program interaction between platforms.

PLCs can be divided into two rough groups according to their hardware architecture: embedded PLCs (ePLCs) and PC-based PLCs. PC-based PLCs offer more user-oriented tools [30], while ePLCs are cheaper and use less power. Given that user-oriented tools are not the focal point of BSIS, ePLCs are a more suitable choice. Plus, the great paradigm shift from PLCs to ePLCs is resulting in some great advances in multiprocessing [32, 33], customized compilation tools [29], support for market-friendly embedded hardware (e.g., Raspberry-Pi), and more. Some of this market-friendly embedded hardware can be used with IoT platforms [34, 35].

2.3. Base Station Management and BSIS. Despite all the advantages of PLCs mentioned above, each is based on a single PLC. But, in base station management, some emphasis needs to be placed on seamless integration between the edge and the cloud to ensure that the edge nodes are responsive, reliable, and scalable.

Responsiveness is reflected in high concurrency and the elasticity of the computing power [4]. Here, reprogramming and real-time tasks must be shifted to the edge where response times can be reduced to within one cycle (e.g., 1 ms).

Reliability means the system is trustworthy and consistently performs well. In the case of base station systems, the reliability of the interactions between the edge nodes and the cloud is important and significant. The guaranteed reliability of PLCs has been formally verified in several studies (e.g., [26, 27, 36]) and, when combined with the above-mentioned advantages, ePLCs make a very robust reliable choice as an edge node. As an added benefit, shifting critical tasks to the edge nodes can also increase safety.

Scalability is reflected in the system's potential for expansion and how it integrates with other systems. Semantic technologies and methods, the capacity for dynamic programming, and compliance with standards must be considered to ensure good scalability. Following [37, 38], we have selected extensible markup language (XML) as the semantic interaction language due to its wide-spread use. As one of the main contributors to scalability, BSIS does support dynamic programming of the edge nodes. Lastly, BSIS complies with the EC61131-3 and PLCopen standards [3, 31].

3. BSIS: A Base Station IoT System

BSIS's main purpose is to store real-time and historical data, manage alarms, and maintain the edge nodes. We have ignored business services like graph analysis and reporting, on-site visualization tools, and order management in this paper, and, instead, have solely focused on an academic analysis of edge node reprogramming and real-time task execution.

The underlying premise of BSIS is that every edge node can and must be able to perform many tasks. Some typical tasks are listed below; most are associated with alarms.

(1) *Core tasks*: the basic functions every edge node must be able to perform.

(2) *Alarm filter (AF)*: where edge nodes read a filter file from a local database and filter the alarms according to the rules in the file. For example, if an alarm has already been signaled by another type of alarm, it is not necessary to report the current status as a separate issue.

(3) *Combined alarms (CA)*: where the data from several sensors is combined and assessed against an alarm threshold so as to reduce the number of alarms. The need for an alarm is determined by several sensors and a logic program.

(4) *Linked tasks (LT)*: where edge nodes directly perform some actions based on an input, for example, taking a photo of the environment if a smoke alarm sounds.

These last three types of tasks are all typical reprogramming and real-time tasks. AF and CA are designed to reduce the number of alarms to a practically manageable level. Linked alarms meet the requirement for real-time responses, which cloud platforms cannot provide.

3.1. The BSIS System Structure. As shown in Figure 1, the system consists of the edge, network, and cloud, which is common for IoT platform [39]. The edge and cloud are connected by a mixed 3G and 4G network.

Edges. The edge nodes are ePLCs and can collect data either directly from the sensors or from the devices using an appropriate communications protocol. The sensors could be measuring temperature, humidity, smoke, infrared light, or vibrations, and the devices could be heat exchange systems, inverters, standard or uninterruptible power systems, ammeters, door systems, ventilation systems, air conditioners, generators, or battery systems.

Cloud Server. The cloud server has basic functions that cover all four types of tasks. Each task corresponds to a unique static program.

Therefore, let BSIS be defined as a 4-tuple set S:

$$\begin{aligned} S &= \{C, P, E, I\}, \\ p^m &\in P, \quad m \le M_{max} \in \mathbb{Z}, \\ e^n &\in E, \quad n \le N_{max} \in \mathbb{Z}, \\ I &= \{\gamma_e, \gamma_c, \lambda_e, \lambda_c, \alpha, \beta\} \end{aligned} \tag{1}$$

where C is the cloud server that produces requests and responds to the edge nodes, P is the finite set of task programs,

Figure 1: The BSIS architecture. BSIS mainly consists of the edge, the network, and the cloud. The edge and cloud are connected by a mixed 3G and 4G network. The cloud part offers basic functions that support many tasks. The ePLCs operate as the edge node and can either collect data from the sensors directly or from the controllers using different communication protocols.

p^m is the mth task program, M_{max} is the total number of tasks, and p^0 represents the basic tasks that should be installed in all edge nodes. E is a finite set of edge nodes, e^n is the nth edge node, N_{max} is the number of edge nodes, e_m^n denotes that the edge node en contains the mth task, and I is the finite set of interaction commands. I includes the edge's request command γ_e, the cloud's request command γ_c, the edge node's response to the cloud λ_e, the cloud's response to the edge node λ_c, the data collection command from the sensors and devices α, and the edge's command to the devices β.

The finite set of interaction commands I is the key to BSIS. It is divided into two parts: (a) the back and forth interactions between the edge nodes and the field sensors/devices (α and β); and (b) the requests made from the edge to the cloud and from the cloud to the edge (γ_e and γ_c) and the corresponding response messages in XML from the edge to the cloud and from the cloud to the edge (λ_e or λ_c). The content of α and β depends on the connected sensors and devices.

Table 1 lists the requests, γ_e and γ_c, and Table 2 shows an example of a data response message by an edge node.

3.2. Edge Computing System. In BSIS, the edge nodes are used to collect heterogeneous data from the devices and sensors and to interact with the cloud. As shown in Figure 2, the software architecture of the ePLCs has a three-layer software structure, which comprises the cloud layer, a flexible layer, and the communication layer.

Table 1: Requests.

γ	Instruction
$\gamma_{e,1}$	Register Request
$\gamma_{e,2}$	Alarm Report Request
$\gamma_{e,3}$	Program Update Request
$\gamma_{c,1}$	Real-time Data Request
$\gamma_{c,2}$	Historical Data Request
$\gamma_{c,3}$	Reading Threshold Request
$\gamma_{c,4}$	Writing Threshold Request
$\gamma_{c,5}$	Edge Node Information Request
$\gamma_{c,6}$	Writing Edge Node Information Request
$\gamma_{c,7}$	Reading Data with FTP Request
$\gamma_{c,8}$	Writing Data with FTP Request
$\gamma_{c,9}$	Clock Synchronization Request

The cloud layer is responsible for interfacing with the cloud. It contains two interfaces, which are the B interface for normal interactions and the M interface for maintenance.

The flexible layer is where nodes are reprogrammed or even changed automatically by a dynamic compilation function in the cloud. This layer mainly consists of types of applications (i.e., e^n). Real-time tasks also run in this layer.

The communication layer connects the devices and sensors with the edge nodes. Almost all communication protocols are supported (e.g., CAN, TCP, UDP, FTP, and

Table 2: An example of a data request by an edge node.

A XML message to report an alarm from an edge node to the cloud

```
<?xml version="1.0"> encoding="UTF-8"
<Request>
  <PK_Type>
    <Name>SEND_ALARM</Name>
    <Code>501</Code>
  </PK_Type>
  <Info>
    <Values>
      <TAlarmList>
        <TAlarm>
          <SerialNo>0012345678</SerialNo>
          <DeviceID>11010110100001</DeviceID>
          <DeviceCode>11010110100001</DeviceCode>
          <AlarmTime>2018-01-04 12:01:31</AlarmTime>
          <FsuId>10024</FsuId>
          <FsuCode>11010110100001</FsuCode>
          <Id>0430101001</Id>
          <AlarmLevel>Level 2</AlarmLevel>
          <AlarmFlag>start</AlarmFlag>
          <AlarmDesc>Low pressure(46.1V)</AlarmDesc>
        </TAlarm>
      </TAlarmList>
    </Values>
  </Info>
</Request>
```

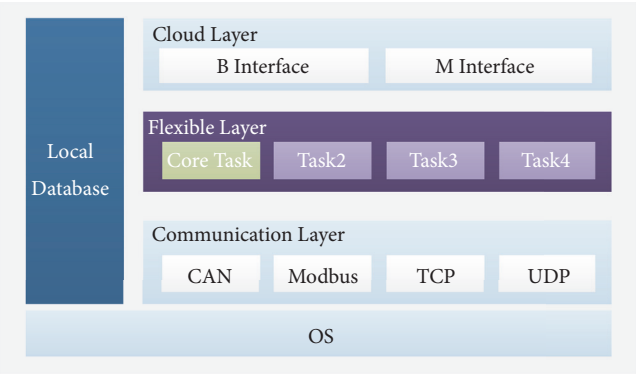


Figure 2: The software architecture of ePLC which encompasses the cloud layer, the flexible layer, and the communication layer all based on the operating system (OS). There is also a local database.

Modbus/RTU), but can be trimmed or changed by the cloud server according to the configuration file.

The underlying operating system provides basic functions.

3.2.1. ePLC Resource Allocation. The RAM available to the ePLC, i.e., its resources, is defined as

$$RE = \{D, M, X, Y, T, C\}, \tag{2}$$

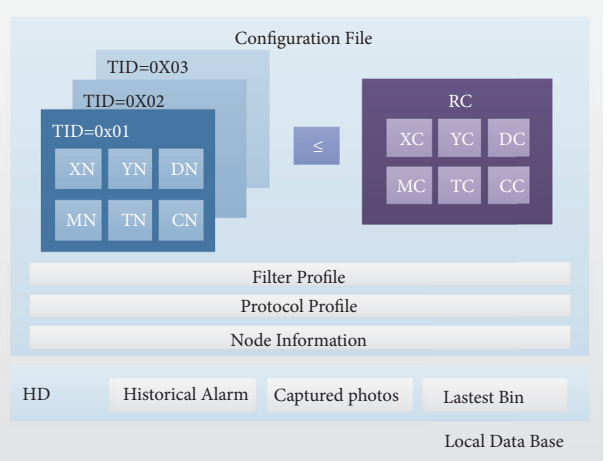


Figure 3: Contents of the configuration file in the local database, which consists of node information, resource constraints, application descriptions, historical data, protocol profiles, and filter profiles.

where RE is composed of RAM addresses with one byte as the smallest unit: D refers to data cell, M to the intermediate cell, X to the input cell, Y to the output cell, T to the timer cell, and C to the counter cell.

3.2.2. Local Data Base. Application programs are independent of each other in the flexible layer. Every application has its own parameter area. A record of its required resources is kept in a configuration file, and a local database stores a limited amount of historical data along with the configuration file (CF), as shown in Figure 3. The CF is described as follows:

$$\begin{aligned} CF &= \{NI, RC, AP, HD, FP, PP\}, \\ RC &= \{XC, YC, DC, MC, TC, CC\}, \\ AP &= \{TID, XN, YN, DN, MN, TN, CN\}, \\ HD &= \{HA, CP, LB\}, \\ PP &= \{PID, EID\}, \end{aligned} \tag{3}$$

where NI is the node information and RC is the set of resource constraints on RE, i.e., the X constraint (XC) and the Y constraint (YC). AP is the set of application descriptions, which contains the task ID (TID) and the required number of resources (X, Y, D, M, T, C). HD is the set of historical data and contains historical alarms (HA), captured photos (CP), and the latest bin (LB). FP denotes the filter profile and PP is the set of protocol profiles, which contains the protocol IDs (PID) and the connected device IDs (EID). The PIDs are used to communicate with the connected device.

Thus, the *ePLC* works as follows:

$$\left[\gamma_e^n, \lambda_e^n, \beta^n\right] = e^n\left(\gamma_c^n, \lambda_c^n, \alpha^n, \delta_\gamma, \delta_\lambda, \delta_\beta\right). \tag{4}$$

Here, every e^n contains several tasks. α^n is the input from sensors and controllers, δ_λ is a condition that produces a

```
Input: r^n_{e,3}
if J(r^n_{e,3}.s) == Flase then
    return;
end
for it=0;it< r^n_{e,3}.N_p;it + + do
    if r^n_{e,3}.version[it]==Null ||
      r^n_{e,3}.version[it]!=Newest then
        compiled = 1;
    end
    P_id[it] = r^n_{e,3}.id[it];
    if it > 0 then
        P_s[it] = Σ_{i=0}^{it-1} r^n_{e,3}.s_i;
    end
end
if compiled == 1 then
    callPLCCompiler(P_id, P_s);
end
```

Algorithm 1: Compilation $\mathscr{H}$.

request, i.e., CF is updated, δ_γ is a condition that triggers the transfer of monitoring data to the cloud, and δ_β is a condition that outputs actions to a controller. As an example, if δ_γ is true, the ePLC will send a request to the cloud to update its program. It is worth noting that each condition and corresponding action is independent.

3.3. Cloud Management. The cloud platform offers the conventional advantages, i.e., flexibility and elastic capacity for computation and storage. We have designed BSIS to operate on a standard platform-as-a-service (PaaS) structure. The reprogramming and real-time tasks are stored on the cloud platform, and every program has a corresponding XML file. When a request is received from an edge node, the cloud platform will respond with $\gamma_{e,3}$, then execute a compilation process and send λ_c to the edge node.

The compilation process is shown in Algorithm 1 and described in detail below. Resource assessment: Compilation terminates if the resource assessment fails. The function that determines whether a request will exceed its resource constraints is defined as

$$\mathscr{J}(s) = \sum_{i=1}^{n} s_{ij}^{l} < s_{jmax}^{l} \quad s.t.\ \forall j = 1, \ldots, 6,\ \ j \in \mathbb{Z}^{+}, \tag{5}$$

where s is the resource matrix and i is the ePLC ID. The rows in s^l are the tasks, and the columns contain the required number of resources (D, M, X, Y, T, C) to complete that task. s_{jmax} represents the resource constraints for D, M, X, Y, T, C in turn.

Link Programs. The cloud checks the program version number according to the program IDs r_e^n in the request to determine whether the program is null or old. If the conditions are satisfied, the compilation flag is set to 1, and the program IDs are stored in another array $P_{id}[it]$. The start-resource address of every program is then calculated and stored in $P_s[it]$.

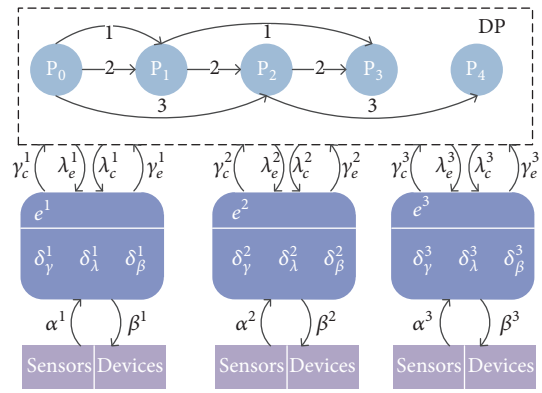


Figure 4: The BSIS model process. λ is the response, and γ is the request. α and β are the input and output of the ePLC, respectively. Ps is the set of all programs in the cloud. δs is the set of conditions.

Call Compiler. If the compilation flag equals 1, the PLC compiler is called according to P_{id}, P_s.

Hence, the cloud process can be described as

$$[\gamma_c^n, \lambda_c^n] = C\left(\lambda_e^n, \mathscr{H}\left(\gamma_{e,3}^n\right), \delta_\gamma, \delta_{op}\right), \quad l \le L \in \mathbb{Z}, \tag{6}$$

where $\mathscr{H}$ is the compilation according to γ_e^n, δ_γ is the condition that produced the request to e^n, and δ_{op} is an option for human intervention. If δ_{op} is true, λ_c will not be sent to the edge node. .

3.4. The Execution Process. Equations (1), (4), and (6) define the BSIS model, and the execution process is illustrated in Figure 4. The edge nodes are crucial because reprogramming and real-time tasks are performed locally according to the input α. Further, the edge nodes can control devices, such as cameras. They directly store some information, such as filtered alarms, and they interact with the cloud using γ and λ including conditions, i.e., δ_γ, δ_λ, and δ_β.

Specifically, $\gamma_{e,3}$ is used to update the edge node's programming. The cloud compiles the task programs according to the request $\gamma_{e,3}$ using Algorithm 1, which results in $(\mathscr{H})$ and sends λ_c to the edge node. Once the B interface in the ePLC receives λ_c, it will update its own bin file in the flexible layer and then hot restart.

4. Performance Analysis

To analyze the performance of BSIS, we constructed a proxy of a base station management system. As shown in Table 3, this BSIS model consists of five servers. The servers sit on the Ali Cloud, and a web server provides various business tasks to users. The task engine server is responsible for task management, and the MySQL server hosts the database. The MQTT server interacts with the edge nodes directly. The VPN server provides a private connection to managers and edge nodes. The ePLCs are IEC61131-3-compliant [40] and are composed of a master CPU (STM32F207VCT6) and a slave (MX283). The master CPU has a frequency of 454 MHz with

Table 3: System configuration.

Ali Cloud	Server Configuration
Web Server	4, 8, 200, 4∗
Task Engine Server	4, 8, 200, 2
MySQL Server	4, 8, 200, 10
MQTT Server	4, 16, 500, 10
VPN Server	2, 8, 200, 10

∗Description: 4,8,200,4 means 4 cores, 8-GB RAM, 200-GB disk, 4-Mbps bandwidth, respectively.

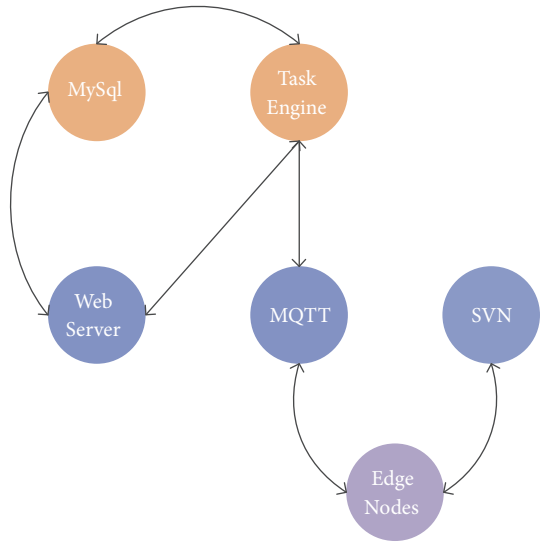


Figure 5: Servers in the cloud and their relationships. The servers include a task engine, an MQTT server, a MySQL server,a web server, and an SVN server.

128MB RAM, and the slave CPU has 120MHZ with 128KB RAM. The local database has a capacity of 10 MB.

Figure 5 shows the relationship between the servers and edge nodes. The task engine interacts with the edge nodes through the MQTT server and, once the data is received, the task engine stores it in the MySQL database. Users are able to communicate with the task engine server or the MySQL database through the web server. Under special circumstances, managers can directly access the edge nodes through the VPN server. The web services and the VPN server are ancillary to this paper since our focus here is on reprogramming and real-time tasks. Hence, no detailed explanations of these components were given in Section 3, and they are only included in the following analyses as comparisons to the edge node response times.

To conduct our analyses, we collected alarm information from 1000 accessed ePLCs for the one-month period of January 2018. The information comprised the dispatch records of maintenance personnel to diagnose the reason for an ePLC being offline in 175 cases. We included the data in our dataset if the ePLC was offline for more than three days or experienced downtime 10 or more times in one day.

Table 4: Combined alarm.

CA	Abbreviation	*rn*
AC input power outage of power supply	AIPOP	3
Rectifier module fault	RMF	47
Low DC output voltage	LDOV	1
Lightning protection device failure, distribution	LPDFD	1
Lightning protection device failure, power supply	LPDFP	1
High AC voltage	HAV	3
Low AC voltage	LAV	3
AC phase loss	APL	3
AC input power outage of intelligent meter	AIPOM	3
High DC output voltage	HDOV	3
Door open overtime of intelligent access	DOOA	1
Door open overtime of non intelligent access	DOONA	1

4.1. Scalability Analysis. As mentioned in Section 2, in base station management, scalability is reflected in the ability of an edge node to be reprogrammed or to perform real-time tasks. In BSIS, the reprogramming and real-time tasks mainly include CA, AF, and LT. However, operators could add any other reprogramming tasks to the cloud server.

Figure 6(a) lists the type and number of tasks in our analysis. There was no record of any AF tasks. Therefore, only CA and LT tasks are considered in these analyses. Figure 6(b) shows the percentage of normal alarms, CAs, and LT alarms based on six months worth of data at 88%, 11%, and 1%, respectively. There was only a relatively small proportion of linked tasks, but all of them are urgent and need to be performed in real-time (e.g., taking a photo after a smoke alarm).

The full name, abbreviation, and number of relevant alarms (*RN*) for the CAs are listed in Table 4. We use ca_i to denote the ith number of CAs and rn_i to denote the ith number of relevant alarms. Note that there are different logic processes for the different types of CAs and their relevant alarms, so we assumed the alarm reduction ratio to be 50%. ta denotes the total number of current alarms (CT), while $\widehat{ca}$ denotes the total number of alarms where a CA was not used (PT). The total reduction in the number of relevant alarms where a CA was not used (CARA) is denoted as $\widehat{ra}$. Hence, $\widehat{ca}$ is defined as

$$\widehat{ca} = \sum_{i=1}^{N_{ca}} ca_i * \frac{rn_i}{2}, \tag{7}$$

where the N_{ca} is the number of CA types that occurred.

Then, $\widehat{ta}$ can be ascertained from

$$\widehat{ta} = ta + \sum_{i=1}^{N_{ca}} ca_i \left(\frac{rn_i}{2} - 1\right). \tag{8}$$

And CARA can be ascertained from $\widehat{ra} = \sum_{i=1}^{N_{ca}} ca_i(rn_i/2 - 1)$.

Applying (7) and (8) to the number and types of CA alarms that occurred in January 2018 only shown in Figure 7(a), we find an overall 60% reduction in alarms.

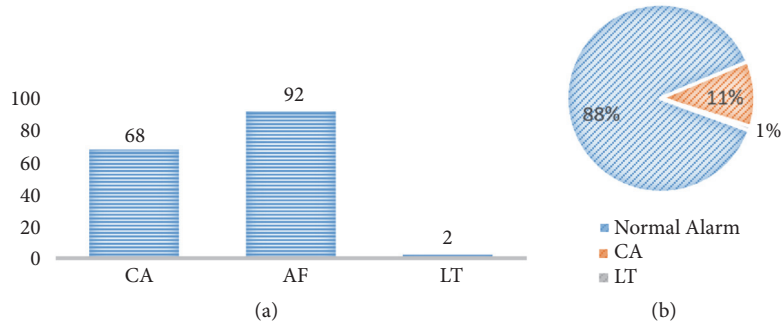


Figure 6: (a) The left side is the Number of CA, AF, and LT tasks. (b) shows the ratio of normal alarms, CA, and LT.

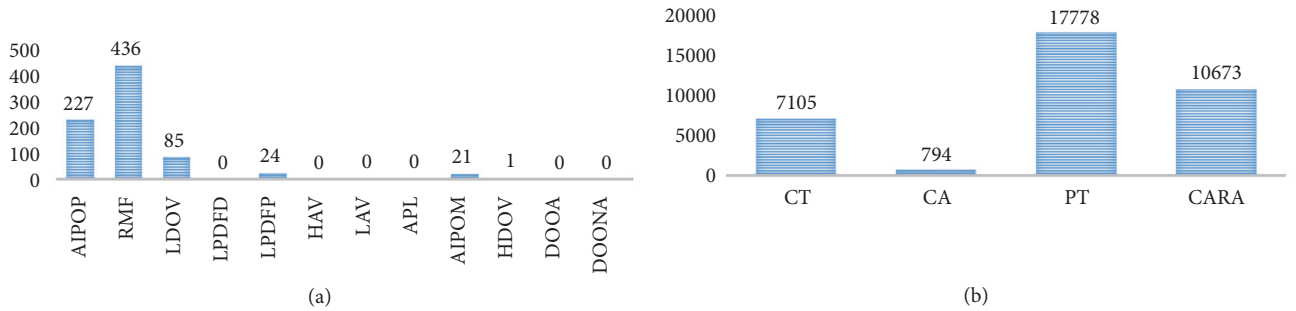


Figure 7: (a) is the occurring alarm number of the different CAs. (b) is the number of CT, CA, PT, and CARA.

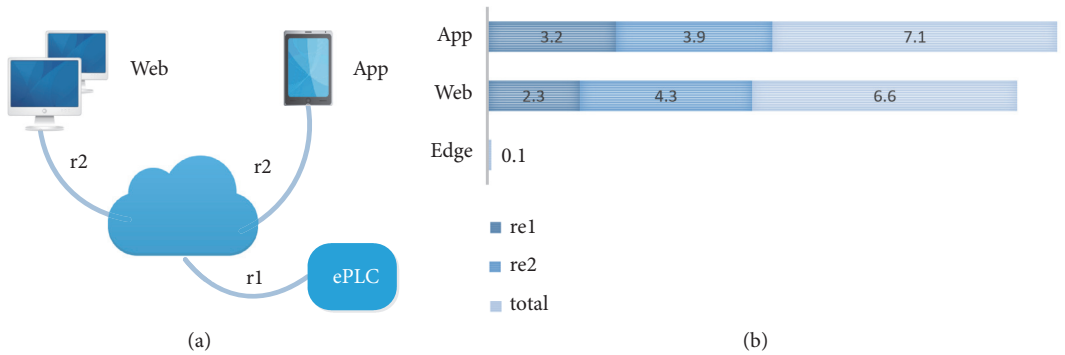


Figure 8: (a) depicts the meaning of r1 and r2. (b) is every stage response time of App, Web, and Edge.

4.2. Response Time Analysis. Our second analysis concerns response times. We compared the response times from the website (Web) and the mobile application (App) with the response times from the edge nodes. The results, averaged over 1000 runs, are shown in Figure 8, divided into two stages. $r1$ is the response time from the edge nodes to the cloud, and $r2$ is the response time from the cloud to the website or mobile application. The edge response times ($r1$) fell within a span of several cycles to less than nearly 0.1 s overall because the ePLCs perform the functions locally. In contrast, the response times between the cloud and the website/app ($r1$) were around 2.3 s and 3.2 s respectively. The mean response times between the cloud and the website/app ($r2$) were 3.9 s and 4.3 s, respectively, largely because wired network speeds are faster than cellular network speeds.

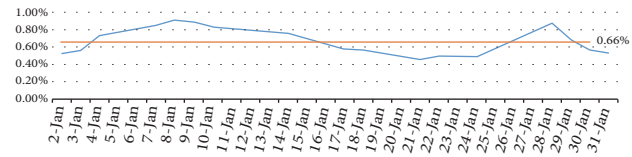


Figure 9: The everyday offline ratio in January 2018. The mean offline ratio is 0.66% which is very low.

4.3. Reliability Analysis. To conduct the reliability analysis, we measured the total downtime of the ePLCs for January 2018. As shown in Figure 9, the average downtime ratio was 0.66%. From this, we can conclude that BSIS has high reliability in terms of uptime. Further, we tracked the reasons for downtime for the 175 cases of offline ePLCs. The reasons

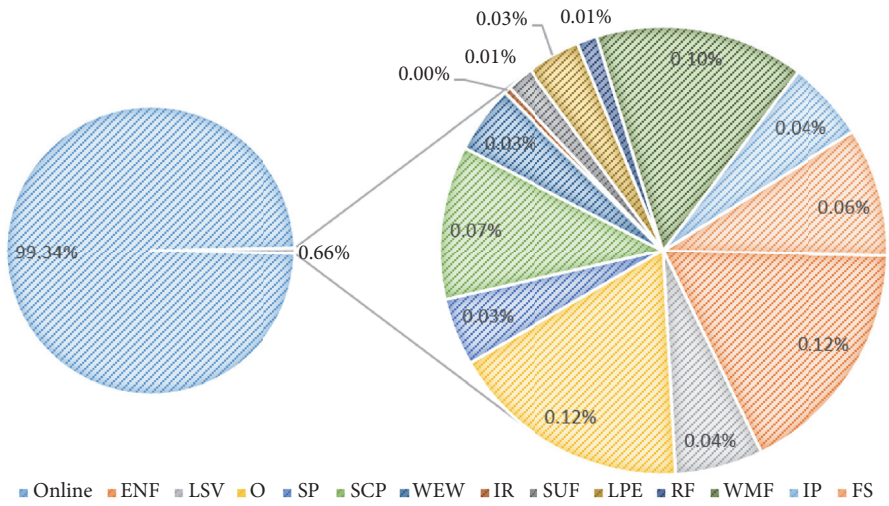


Figure 10: Offline ratio and offline reason ratio.

were distilled into categories: edge node faults (ENF) 36; outdated software (LSV) 13; signal problems (SP) 10; SIM card problems (SCP) 23; wiring errors in the wireless module (WEW) 10; duplicate IDs (IR) 1; software update failures (SUF) 4; low power to the edge node (LPE) 8; restart failures (RF) 3, wireless module faults (WMF) 31; IP problems (IP) 12; fake stations (FS) 19; and "other"(O) 36. Figure 10 illustrates these problems with the corresponding ePLC downtime they caused, assuming these reasons account for all 0.66% of the downtime. This data will be used to inform future research.

5. Conclusion

In this paper, we explained the problems associated with using cloud computing as a platform for base station management systems. Our solution in response is an edge-cloud IoT computing system, called BSIS, that allows ePLCs to act as edge nodes to perform specific tasks locally. BSIS has several advantages. The edge nodes are reprogrammable and have much lower real-time response rates than a cloud server, which is particularly beneficial for sounding alarm signals and linking tasks. Further, BSIS seamlessly integrates the cloud and edge components of the system. Analyses of the scalability, responsiveness, and reliability of BSIS indicate a 60% reduction in the number of alarms, a potential edge response time of less than 0.1s, and a downtime ratio of 0.66%.

In future work, we will explore the potential of integrating artificial intelligence into the system architecture to further improve the performance of BSIS.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by a grant from the National Natural Science Foundation of China (no. U1609211) and the Science and Technology Program of Zhejiang Province (no. 2018C04001).

References

[1] W. Yu, F. Liang, X. He et al., "A survey on the edge computing for the internet of things," *IEEE Access*, vol. 6, pp. 6900–6919, 2017.

[2] H. El-Sayed, S. Sankar, M. Prasad et al., "Edge of things: The big picture on the integration of edge, IoT and the cloud in a distributed computing environment," *IEEE Access*, vol. 6, pp. 1706–1717, 2017.

[3] *Programmable Controllers - Part 3: Programming Languages*, International Electrotechnical Commission, 2013, https://webstore.iec.ch/publication/4552.

[4] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[5] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[6] X. Chen, Q. Shi, L. Yang, and J. Xu, "ThriftyEdge: resource-efficient edge computing for intelligent IoT applications," *IEEE Network*, vol. 32, no. 1, pp. 61–65, 2018.

[7] S. Li, N. Zhang, S. Lin et al., "Joint admission control and resource allocation in edge computing for internet of things," *IEEE Network*, vol. 32, no. 1, pp. 72–79, 2018.

[8] L. Yin, J. Luo, and H. Luo, "Tasks scheduling and resource allocation in fog computing based on containers for smart manufacturing," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4712–4721, 2018.

[9] J.-S. Fu, Y. Liu, H.-C. Chao, B. K. Bhargava, and Z.-J. Zhang, "Secure data storage and searching for industrial iot by integrating fog computing and cloud computing," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4519–4528, 2018.

[10] D. A. Chekired, L. Khoukhi, and H. T. Mouftah, "Industrial IoT data scheduling based on hierarchical fog computing: a key for enabling smart factory," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4590–4602, 2018.

[11] T. Suganuma, T. Oide, S. Kitagami, K. Sugawara, and N. Shiratori, "Multiagent-based flexible edge computing architecture for IoT," *IEEE Network*, vol. 32, no. 1, pp. 16–23, 2018.

[12] G. Jia, G. Han, H. Rao, and L. Shu, "Edge computing-based intelligent manhole cover management system for smart cities," *IEEE Internet of Things Journal*, 2017.

[13] G. Xu, W. Yu, D. Griffith, N. Golmie, and P. Moulema, "Toward integrating distributed energy resources and storage devices in smart grid," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 192–204, 2017.

[14] P. Pace, G. Aloi, R. Gravina, G. Caliciuri, G. Fortino, and A. Liotta, "An edge-based architecture to support efficient applications for healthcare industry 4.0," *IEEE Transactions on Industrial Informatics*, 2018.

[15] J. Lin, W. Yu, X. Yang, Q. Yang, X. Fu, and W. Zhao, "A real-time en-route route guidance decision scheme for transportation-based cyberphysical systems," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, pp. 2551–2566, 2017.

[16] I. Kalamaras, A. Zamichos, A. Salamanis et al., "An interactive visual analytics platform for smart intelligent transportation systems management," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 2, pp. 487–496, 2018.

[17] M. Metzger and G. Polaków, "A survey on applications of agent technology in industrial process control," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 4, pp. 570–581, 2011.

[18] J. Qin, Q. Ma, Y. Shi, and L. Wang, "Recent advances in consensus of multi-agent systems: a brief survey," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 6, pp. 4972–4983, 2017.

[19] Y. Rizk, M. Awad, and E. W. Tunstel, "Decision making in multiagent systems: a survey," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 10, no. 3, pp. 514–529, 2018.

[20] M. S. Familiar, J. F. Martínez, and L. López, "Pervasive smart spaces and environments: a service-oriented middleware architecture for wireless ad hoc and sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2012, Article ID 725190, 11 pages, 2012.

[21] T. Cucinotta, A. Mancina, G. F. Anastasi et al., "A real-time service-oriented architecture for industrial automation," *IEEE Transactions on Industrial Informatics*, vol. 5, no. 3, pp. 267–277, 2009.

[22] A. Girbea, C. Suciu, S. Nechifor, and F. Sisak, "Design and implementation of a service-oriented architecture for the optimization of industrial applications," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 185–196, 2014.

[23] M. Abdur Rahman, M. Shamim Hossain, E. Hassanain, and G. Muhammad, "Semantic multimedia fog computing and IoT environment: sustainability perspective," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 80–87, 2018.

[24] L. Hernandez, H. Cao, and M. Wachowicz, "Implementing an edge-fog-cloud architecture for stream data management," in *Proceedings of the IEEE Fog World Congress (FWC '17)*, pp. 1–6, IEEE, Santa Clara, Calif, USA, October 2017.

[25] C.-Y. Chang, "Adaptive fuzzy controller of the overhead cranes with nonlinear disturbance," *IEEE Transactions on Industrial Informatics*, vol. 3, no. 2, pp. 164–172, 2007.

[26] Y. Jiang, H. Zhang, H. Liu et al., "System reliability calculation based on the run-time analysis of ladder program," in *Proceedings of the 9th Joint Meeting on Foundations of Software Engineering*, pp. 695–698, ACM, Saint Petersburg, Russia, August 2013.

[27] Y. Jiang, H. Zhang, X. Song et al., "Bayesian-network-based reliability analysis of PLC systems," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 11, pp. 5325–5336, 2013.

[28] S. Dominic, Y. Lohr, A. Schwung, and S. X. Ding, "PLC-based real-time realization of flatness-based feedforward control for industrial compression systems," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 2, pp. 1323–1331, 2017.

[29] H. Wu, Y. Yan, D. Sun, and R. Simon, "A customized real-time compilation for motion control in embedded PLCs," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 2, pp. 812–821, 2019.

[30] S. Hossain, M. A. Hussain, and R. B. Omar, "Advanced control software framework for process control applications," *International Journal of Computational Intelligence Systems*, vol. 7, no. 1, pp. 37–49, 2014.

[31] Function blocks for motion control version 1.1, PLCopen Technical Commitee 2, http://www.plcopen.org/pages/tc2_motion_control/, 2005.

[32] Z. Hajduk, B. Trybus, and J. Sadolewski, "Architecture of FPGA embedded multiprocessor programmable controller," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 5, pp. 2952–2961, 2015.

[33] M. Chmiel, J. Kulisz, R. Czerwinski, A. Krzyzyk, M. Rosol, and P. Smolarek, "An IEC 61131-3-based PLC implemented by means of an FPGA," *Microprocessors and Microsystems*, vol. 44, pp. 28–37, 2016.

[34] B. Chen, J. Wan, A. Celesti, D. Li, H. Abbas, and Q. Zhang, "Edge computing in IoT-based manufacturing," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 103–109, 2018.

[35] R. Usamentiaga, M. A. Fernandez, A. Fernandez, and J. L. Carus, "Temperature monitoring for electrical substations using infrared thermography: architecture for Industrial Internet of Things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 12, pp. 5667–5677, 2018.

[36] B. F. Adiego, D. Darvas, E. B. Viñuela et al., "Applying model checking to industrial-sized PLC programs," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 6, pp. 1400–1410, 2015.

[37] Y. Doi, Y. Sato, M. Ishiyama, Y. Ohba, and K. Teramoto, "XML-less EXI with code generation for integration of embedded devices in web based systems," in *Proceedings of the 3rd International Conference on the Internet of Things (IOT '12)*, pp. 76–83, Wuxi, Jiangsu Province, China, October 2012.

[38] A. Zanella, N. Bui, A. P. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014.

[39] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on internet of things: architecture, enabling technologies, security and privacy, and applications," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125–1142, 2017.

[40] Y. Yan and H. Zhang, "Compiling ladder diagram into instruction list to comply with IEC 61131-3," *Computers in Industry*, vol. 61, no. 5, pp. 448–462, 2010.

Hindawi
Complexity
Volume 2019, Article ID 4829164, 18 pages
https://doi.org/10.1155/2019/4829164

Research Article

A Novel Index Method for K Nearest Object Query over Time-Dependent Road Networks

Yajun Yang[1,2] **Hanxiao Li**[1] **Junhu Wang**[3] **Qinghua Hu**[1] **Xin Wang**[1] **and Muxi Leng**[1]

[1]*College of Intelligence and Computing, Tianjin University, China*
[2]*State Key Laboratory of Digital Publishing Technology, China*
[3]*School of Information and Communication Technology, Griffith University, Australia*

Correspondence should be addressed to Xin Wang; wangx@tju.edu.cn

Received 30 November 2018; Accepted 28 January 2019; Published 24 February 2019

Guest Editor: Jiajie Xu

K nearest neighbor (*k*NN) search is an important problem in *location-based services* (LBS) and has been well studied on static road networks. However, in real world, road networks are often time-dependent; i.e., the time for traveling through a road always changes over time. Most existing methods for *k*NN query build various indexes maintaining the shortest distances for some pairs of vertices on static road networks. Unfortunately, these methods cannot be used for the time-dependent road networks because the shortest distances always change over time. To address the problem of *k*NN query on time-dependent road networks, we propose a novel voronoi-based index in this paper. Furthermore, we propose a novel balanced tree, named V-tree, which is a secondary level index on voronoi-based index to make our querying algorithm more efficient. Moreover, we propose an algorithm for preprocessing time-dependent road networks such that the waiting time is not necessary to be considered. We confirm the efficiency of our method through experiments on real-life datasets.

1. Introduction

With the rapid development of mobile devices, *k* nearest neighbor (*k*NN) search on road networks has become more and more important in *location-based services* (LBS)[1–4]. Given a query location and a set of objects (e.g., restaurants) on a road network, it is to find *k* nearest objects to the query location. *k*NN search problem has been well studied on static road networks. However, road networks are essentially *dynamic complex networks* but not static in real world [5–8]. The "dynamic" means that the traveling time on road network is always time-dependent. For example, the Vehicle Information and Communication System (VICS) and the European Traffic Message Channel (TMC) are two transportation systems, which provide real-time traffic information to users. Such road networks are time-dependent; i.e., travel time for a road varies with taking "rush hour" into account.

The existing works propose various index techniques for answering *k* nearest object query on road networks. The main idea behind these indexes is to partition the vertices into several clusters, and then the clusters are organized as a voronoi diagram or a tree (e.g., R-tree, G-tree). All these methods precompute and maintain the shortest distances for some pairs of vertices to facilitate *k*NN query. Unfortunately, these indexes cannot be used for time-dependent road networks. The reason is that the minimum travel time between two vertices often varies with time. For example, u and v are in the same cluster for one time period but they may be in two distinct clusters for another time period because of the minimum travel time varying with time. Therefore, the existing index techniques based on the static shortest distance cannot handle the case that the minimum travel time is time-dependent. Moreover, the waiting time is allowed on time-dependent road networks; i.e., someone can wait a time period to find another faster path. When the waiting time is considered, it is more difficult to build an index for *k*NN query by existing methods because it is difficult to estimate an appropriate waiting time for precomputing the minimum travel time between two vertices.

Recently, there are some works about kNN query on time-dependent graphs [9–12]. Most of these works utilize A$*$ algorithm to expand the road networks by estimating an upper or lower bound of travel time. There are two main drawbacks of these methods. First, in these works, the FIFO (first in first out) property is required for the networks and the waiting time is not allowed. Second, the indexes proposed by these works are based on the estimated value of travel time. However, these indexes cannot facilitate query effectively for large networks because the deviations are always too large between the estimated and the actual travel time.

In this paper, we study k nearest object query on time-dependent road networks. A time-dependent road network is modeled as a graph with time information. The weight of every edge is a time function $w_{i,j}(t)$ which specifies how much time it takes to travel through the edge (v_i, v_j) if departing at time point t. The main idea of our method is to precompute minimum travel time functions (or mtt-function for short) instead of concrete values for some pairs of vertices and then design a "*dynamic*" voronoi-based index based on such functions. Here "dynamic" means that in a time-dependent network it can be easily decided which cluster a vertex should be in for any given time point t. Furthermore, a secondary level index is built on voronoi-based index, which makes our querying algorithm more "smart" for k nearest objects searching; that is, the k nearest objects will be searched and the others will be filtered as early as possible. Different to previous works, our index can facilitate query effectively for large networks. Moreover, our method does not require the FIFO property for networks and we allow waiting time on every vertex.

The main contributions of this paper are summarized as below. First, we propose an algorithm to process $w_{i,j}(t)$ for every edge such that the waiting time is not necessary to be considered. Let G_T and G_T^* be the original graph and the graph after processing $w_{i,j}(t)$. We can prove that a shortest path with consideration of waiting time on G_T is one-to-one mapped to shortest path without waiting time on G_T^*. Furthermore, we show how to compute the mtt-function for two vertices. Second, we propose a novel voronoi-based index for time-dependent road networks and an algorithm to answer kNN query using our index. Third, we propose a novel balanced tree structure, named V-tree, which can be considered as the secondary level index on voronoi-based index to make k nearest object query more efficient. Finally, we confirm the efficiency of our method through extensive experiments on real-life datasets.

The rest of this paper is organized as follows. Section 2 gives the problem statement. Section 3 decribes how to process $w_{i,j}(t)$ and compute the mtt-function. Section 4 explains how to build the voronoi-based index for time-dependent networks. Section 5 introduces the V-tree and how to answer the k nearest object query using it. The experimental results are presented in Section 6. The related work is in Section 7. Finally, we conclude this paper in Section 8.

2. Problem Statement

Definition 1 (time-dependent road network). A time-dependent road network is a simple directed graph, denoted as $G_T(V, E, W)$ (or G_T for short), where V is the set of vertices; $E \subseteq V \times V$ is the set of edges; and W is a set of nonnegative value function. For every edge $(v_i, v_j) \in E$, there is a time function $w_{i,j}(t) \in W$, where t is a time variable. A time function $w_{i,j}(t)$ specifies how much time it takes to travel from v_i to v_j, if one departs from v_i at time point t.

In this paper, we assume that $w_{i,j}(t) \geq 0$. The assumption is reasonable, because the travel time cannot be less than zero in real applications. Our work can be easily extended to handle undirected graphs. An undirected edge (v_i, v_j) is equivalent to two directed edges (v_i, v_j) and (v_j, v_i), where $w_{i,j}(t) = w_{j,i}(t)$.

There are several works that study how to construct time function $w_{i,j}(t)$, which is always modeled as a piecewise linear function [6, 7, 13] and it can be formalized as follows:

$$w_{i,j}(t) = \begin{cases} a_1 t + b_1, & t_0 \leq t < t_1 \\ a_2 t + b_2, & t_1 \leq t < t_2 \\ \cdots & \\ a_p t + b_p, & t_{p-1} \leq t \leq t_p \end{cases} \tag{1}$$

Given a path p, the travel time of p is time-dependent. In order to minimize the travel time, some waiting time ω_i is allowed at every vertex v_i in p. That is, when arriving at v_i, one can wait a time period ω_i if the travel time of p can be minimized. We use $\mathsf{arrive}(v_i)$ and $\mathsf{depart}(v_i)$ to denote the arrival time at v_i and departure time from v_i, respectively. For each v_i in p, we have

$$\mathsf{depart}(v_i) = \mathsf{arrive}(v_i) + \omega_i \tag{2}$$

Let $p = v_1 \longrightarrow v_2 \longrightarrow \cdots \longrightarrow v_h$ be a given path with the departure time t and the waiting time ω_i for each vertex v_i; then we have

$$\begin{gathered} \mathsf{arrive}(v_1) = t \\ \mathsf{arrive}(v_2) = \mathsf{depart}(v_1) + w_{1,2}(\mathsf{depart}(v_1)) \\ \cdots \\ \mathsf{arrive}(v_h) = \mathsf{depart}(v_{h-1}) + w_{h-1,h}(\mathsf{depart}(v_{h-1})) \end{gathered} \tag{3}$$

Thus the travel time of path p is $w(p) = \mathsf{arrive}(v_h) - t$. Given two vertices v_i and v_j in G_T, the minimum travel time from v_i to v_j with departure time t is defined as $m_{i,j}(t) = \min\{w(p) \mid p \in P_{i,j}\}$, where $P_{i,j}$ is the set of all the paths from v_i to v_j in G_T. Obviously, $m_{i,j}(t)$ is also a function related to the departure time t. We call $m_{i,j}(t)$ the *minimum travel time function* (or mtt-function shortly) from v_i to v_j. Letting $|V|$ be n, in the following, we use $m_{i,n+j}(t)$ to represent mtt-function from a vertex v_i to an object o_j, in order to distinguish from $m_{i,j}(t)$ from v_i to a vertex v_j. Note that an object o_i is also a vertex regarded as v_{n+i} in the network.

There is a special case that the waiting time is only allowed on a subset V_w of V; that is, people are allowed to wait a time period ω_i only when they are at the vertex $v_i \in V_w$. For this case, we only need to consider that the waiting time is always zero, i.e., $\omega_i \equiv 0$, for every vertex $v_i \in V \setminus V_w$. Therefore, the method proposed in this paper can be extended to handle this case easily.

Next, we give the definition of kNN query over time-dependent road networks.

Definition 2 (k nearest objects on time-dependent road networks). Given a time-dependent road network $G_T(V, E, W)$, a set of the objects $O = \{o_1, o_2, \cdots\}$, a query point $v_q \in V$, and a departure time t_d, k nearest object query of v_q is to find a k-size subset $O(v_q) \subseteq O$, such that $m_{q,n+j}(t_d) \geq \max\{m_{q,n+i}(t_d) \mid o_i \in O(v_q)\}$ for every object $o_j \in O \setminus O(v_q)$.

3. Minimum Travel Time Function

We precompute mtt-functions for some pairs of vertices and then build the index to facilitate kNN query over time-dependent road networks. In this section, we first describe how to process the time function $w_{i,j}(t)$ for every edge in G_T such that the waiting time is not necessary to be considered when computing mtt-function and then we explain how to compute mtt-function without waiting time.

3.1. Preprocessing Time Function for Every Edge. Given a path p, the waiting time ω_i is allowed for any vertex $v_i \in p$. However, it is not easy to find an appropriate value of ω_i for every $v_i \in p$ to minimize the travel time of p. In this section, we propose an algorithm to convert time function $w_{i,j}(t)$ to a new function $w^*_{i,j}(t)$ for every edge $(v_i, v_j) \in E$. We call $w^*_{i,j}(t)$ the "no waiting time function" of edge (v_i, v_j) (or nwt-function for short). The waiting time can be considered as zero when nwt-function is used to compute the minimum travel time of path p. The nwt-function $w^*_{i,j}(t)$ is defined by the following:

$$w^*_{i,j}(t) = \min_{\omega_i} \left(\omega_i + w_{i,j}\left(t + \omega_i\right)\right) \tag{4}$$

Note that when the waiting time is only allowed on a vertex subset V_w, we only need to calculate nwt-function $w^*_{i,j}(t)$ for every vertex $v_i \in V_w$. For every vertex $v_i \in V \setminus V_w$, the nwt-function $w^*_{i,j}(t)$ is exactly the time function $w_{i,j}(t)$ because $\omega_i \equiv 0$; that is, anyone is not allowed to wait a time period ω_i at v_i.

The following theorem guarantees that the nwt-function $w^*_{i,j}(t)$ can be used to compute the minimum travel time for any path p in G_T without waiting time.

Theorem 3. *Given two time-dependent graphs $G_T(V, E, W)$ and $G^*_T(V, E, W^*)$, where W^* is the set of* nwt*-functions of all edges in E, for any path p in G_T, the minimum travel time of p in G_T with consideration of waiting time equals the minimum travel time of p in G^*_T without waiting time.*

Proof. Let $p = v_1 \longrightarrow v_2 \longrightarrow \cdots \longrightarrow v_h$ be a given path with the departure time t. ω^*_i is the waiting time on v_i $(1 \leq i \leq h)$ minimizing the travel time of p in G_T. We have $\mathsf{depart}(v_i) = \mathsf{arrive}(v_i) + \omega^*_i$ and $\mathsf{arrive}(v_{i+1}) = \mathsf{depart}(v_i) + w_{i,i+1}(\mathsf{depart}(v_i))$. Similarly, we have $\mathsf{depart}^*(v_i) = \mathsf{arrive}^*(v_i)$ and $\mathsf{arrive}^*(v_{i+1}) = \mathsf{depart}^*(v_i) + w^*_{i,i+1}(\mathsf{depart}^*(v_i))$ for G^*_T. We only need to prove $\mathsf{arrive}(v_h) = \mathsf{arrive}^*(v_h)$. It can be easily proved by induction on v_i. We omit it due to the space limitation. □

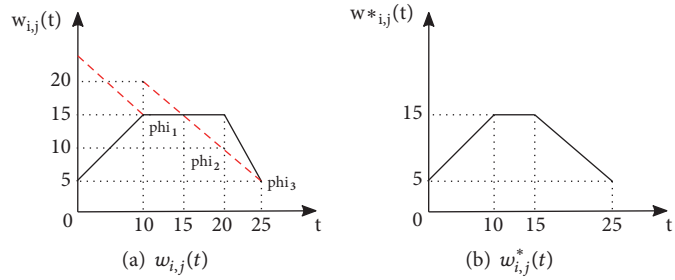
The algorithm to compute nwt-function is shown in Algorithm 1. For every $w_{i,j}(t) \in W$, if $v_i \in V_w$, Algorithm 1 computes $w^*_{i,j}(t)$ backward from $[t_{p-1}, t_p]$ to $[t_0, t_1]$ iteratively. In each iteration, $w^*_{i,j}(t)$ for $t \in [t_{k-1}, t_k)$ is computed. Algorithm 1 first sets $w^*_{i,j}(t)$ as $a^*t + b^*$, where $a^* = -1$ and $b^* = t_k + \phi$. ϕ is the minimum value between $w^*_{i,j}(t_k)$ and $w^-_{i,j}(t_k)$. $w^-_{i,j}(t_k)$ is the left limit value of $w_{i,j}(t)$ on t_k. Note that $w^*_{i,j}(t_k)$ and ϕ have been computed in the last iteration; i.e., the iteration for computing $w^*_{i,j}(t)$ on $[t_k, t_{k+1})$. ϕ is initialized as $w_{i,j}(t_p)$. Next, Algorithm 1 updates $w^*_{i,j}(t)$ as $\min\{w^*_{i,j}(t), w_{i,j}(t)\}$ for $t \in [t_{k-1}, t_k)$ and then ϕ is updated as $\min\{w^*_{i,j}(t_{k-1}), w^-_{i,j}(t_{k-1})\}$. The algorithm terminates when $w^*_{i,j}(t)$ has been computed for $t \in [t_0, t_1)$. Note that if $v_i \notin V_w$, $w^*_{i,j}(t)$ is computed as $w_{i,j}(t)$ immediately.

The time and space complexities analysis for Algorithm 1 are given below. Let n and m be the number of the vertices and edges in G_T, respectively. For every edge (v_i, v_j), Algorithm 1 needs to compute $w^*_{i,j}(t)$ on $[t_{k-1}, t_k)$ iteratively from $k = p$ to 1. For every time interval $[t_{k-1}, t_k)$, $w^*_{i,j}(t)$ can be computed in constant time. Therefore, the time complexity of Algorithm 1 is $O(mp)$. Moreover, Algorithm 1 needs to maintain $w^*_{i,j}(t)$ and then the space complexity is also $O(mp)$.

Example 4. We illustrate how to compute $w^*_{i,j}(t)$ by an example in Figure 1. As the solid black line in Figure 1(a), $w_{i,j}(t)$ is a piecewise linear function:

$$w_{i,j}(t) = \begin{cases} t + 5, & 0 \leq t < 10 \\ 15, & 10 \leq t < 20 \\ -2t + 55, & 20 \leq t \leq 25 \end{cases} \tag{5}$$

In the first iteration, ϕ is initialized as $w_{i,j}(25) = 5$ and then $b^* = 25 + \phi = 30$. As the dashed red line in the right-side of Figure 1(a), we find that $a^*t + b^* = -t + 30$ is always less than $w_{i,j}(t)$ on $[20, 25]$, then $w^*_{i,j}(t) = -t + 30$ for $t \in [20, 25]$, and ϕ is updated as 10. Similarly, in the second iteration, $w^*_{i,j}(t)$ on $[10, 20)$ is computed as $\min\{15, -t + 30\}$, i.e., $w^*_{i,j}(t) = 15$ for $t \in [10, 15)$ and $w^*_{i,j}(t) = -t + 30$ for $t \in [15, 20)$. Then ϕ is updated as $\min\{w^*_{i,j}(10), w^-_{i,j}(10)\} = 15$. In the final iteration, as the dashed red line in the left-side of Figure 1(a), $a^*t + b^* = -t + 25$ is alway larger than $t + 5$ on $[0, 10)$, and we have $w^*_{i,j}(t) =$



(a) $w_{i,j}(t)$ (b) $w^*_{i,j}(t)$

Figure 1: Computing $w^*_{i,j}(t)$.

Input: $G_T(V, E, W)$ and V_w.
Output: W^*.
1: $W^* \longleftarrow \emptyset$;
2: *for* every $w_{i,j}(t) \in W$ *do*
3: *if* $v_i \in V_w$ *then*
4: $\phi \longleftarrow w_{i,j}(t_p), w^*_{i,j}(t_p) \longleftarrow w_{i,j}(t_p)$;
5: *for* $k = p$ to 1*do*
6: $a^* \longleftarrow -1, b^* \longleftarrow t_k + \phi$;
7: $w^*_{i,j}(t) \longleftarrow a^* t + b^*$ for $t \in [t_{k-1}, t_k)$;
8: $w^*_{i,j}(t) \longleftarrow \min\{w^*_{i,j}(t), w_{i,j}(t) \mid t \in [t_{k-1}, t_k)\}$;
9: $\phi \longleftarrow \min\{w^*_{i,j}(t_{k-1}), w^-_{i,j}(t_{k-1})\}$;
10: *else*
11: $w^*_{i,j}(t) \longleftarrow w_{i,j}(t)$;
12: $W^* \longleftarrow W^* \cup \{w^*_{i,j}(t)\}$;
13: *return* W^*

Algorithm 1: Nwt-Function $(G_T(V, E, W), V_w)$.

$t + 5$ for $t \in [0, 10)$. Then $w^*_{i,j}(t)$ is given below and depicted in Figure 1(b)).

$$w^*_{i,j}(t) = \begin{cases} t + 5, & 0 \le t < 10 \\ 15, & 10 \le t < 15 \\ -t + 30, & 15 \le t \le 25 \end{cases} \tag{6}$$

The following theorem guarantees the correctness of Algorithm 1.

Theorem 5. *$w^*_{i,j}(t)$ computed by Algorithm 1 is exactly the* nwt*-function $w^*_{i,j}(t)$ given by (4).*

Proof. We proved it by induction on p.

Basis. We need to prove that $w^*_{i,j}(t)$ on time interval $[t_{p-1}, t_p]$ can be correctly computed by Algorithm 1. First, ω_i can only be zero when $t = t_p$; then we have $w_{i,j}(t_p) = w^*_{i,j}(t_p)$ and $\phi_p = w_{i,j}(t_p)$. Next, we consider the case of $t \in [t_{p-1}, t_p)$.

By the definiton of $w^*_{i,j}(t)$, we have

$$\begin{aligned} w^*_{i,j}(t) &= \min_{\omega_i} \left(\omega_i + w_{i,j}(t + \omega_i)\right) \\ &= \min_{\omega_i} \left(\omega_i + a_p(t + \omega_i) + b_p\right) \\ &= \min_{\omega_i} \left(\left(a_p + 1\right)\omega_i + a_p t + b_p\right) \\ &= \min_{\omega_i} \left(\left(a_p + 1\right)\omega_i + w_{i,j}(t)\right) \end{aligned} \tag{7}$$

For $t \in [t_{p-1}, t_p)$, if $a_p \ge -1$, $w^*_{i,j}(t)$ cannot decrease with ω_i increasing. It means that $(a_p + 1)\omega_i + w_{i,j}(t)$ is minimum when $\omega_i = 0$ and then $w^*_{i,j}(t) = w_{i,j}(t)$. If $a_p < -1$, $w^*_{i,j}(t)$ will decrease with ω_i increasing and thus $(a_p + 1)\omega_i + w_{i,j}(t)$ is minimum when $\omega_i = t_p - t$, which is the longest waiting time on v_i for $t \in [t_{p-1}, t_p)$. Then we have

$$w^*_{i,j}(t) = \left(a_p + 1\right)\left(t_p - t\right) + a_p t + b_p = -t + t_p + \phi_p \tag{8}$$

Obviously, $w_{i,j}(t) \le -t + t_p + \phi_p$ when $a_p \ge -1$ and $w_{i,j}(t) \ge -t + t_p + \phi_p$ when $a_p < -1$. Then we have $w^*_{i,j}(t) = \min\{w_{i,j}(t), -t + t_p + \phi_p\}$ for $t \in [t_{p-1}, t_p]$.

Induction. Assuming that the correct $w^*_{i,j}(t)$ can be computed by Algorithm 1 for $t \in [t_k, t_p]$, then we need to prove it also can be correctly computed for $t \in [t_{k-1}, t_k)$. We consider the following two cases: (1) $\omega_i \ge t_k - t$ and (2) $\omega_i < t_k - t$.

For case (1), the departure time $t + \omega_i \in [t_k, t_p]$ because $\omega_i \ge t_k - t$. By the assumption, nwt-function $w^*_{i,j}(t)$ has been correctly computed for $t \in [t_k, t_p]$; then $w^*_{i,j}(t_k)$ is the minimum travel time for edge (v_i, v_j) with departure time t_k. Therefore, $w^*_{i,j}(t)$ for $t \in [t_{k-1}, t_k)$ can be computed by the following:

$$w^*_{i,j}(t) = t_k - t + w^*_{i,j}(t_k) \tag{9}$$

For case (2), because $\omega_i < t_k - t$, then $t + \omega_i \in [t_{k-1}, t_k)$. Similar to the proof of basis, we have

$$w^*_{i,j}(t) = \min\left\{w_{i,j}(t), -t + t_k + w^-_{i,j}(t_k)\right\} \tag{10}$$

Note that when $a_k < -1$, $w^*_{i,j}(t) = -t + t_k + w^-_{i,j}(t_k)$ because $w_{i,j}(t)$ may be noncontinous at t_k. Therefore, we have

$$w^*_{i,j}(t) = \min\left\{w_{i,j}(t), -t + t_k + w^-_{i,j}(t_k), -t + t_k + w^*_{i,j}(t_k)\right\} \quad (11)$$

The proof is completed. □

3.2. Computing Minimum Travel Time Function. We adopt a Dijkstra-based algorithm proposed in [13] to compute mtt-function for two vertices v_i and v_j in G_T. This algorithm is only used in case the waiting time is not allowed. After converting $w_{i,j}(t)$ to nwt-function $w^*_{i,j}(t)$ for every edge in G_T by Algorithm 1, this algorithm can be used for time-dependent graphs with waiting time.

The main idea of this Dijkstra-based algorithm is to refine a function $g_{i,j}(t)$ iteratively for every $v_j \in V$, where $g_{i,j}(t)$ represents the earliest arrival time on v_j if departing from v_i at time point t. In every iteration, algorithm selects a vertex $v_x \in V$ and then refines $g_{i,x}(t)$ by extending a time domain I_x to a larger I'_x, where $I_x = [t_0, \tau_x]$ is a subinterval of the whole time domain T. $g_{i,x}(t)$ is regarded as well refined in I_x if it specifies the earliest arrival time at v_x from v_i for any departure time $t \in I_x$. The algorithm repeats time-refinement process till $g_{i,j}(t)$ of destination v_j has been well refined in the whole time domain T and then mtt-function $m_{i,j}(t)$ can be computed as $m_{i,j}(t) = g_{i,j}(t) - t$. The more details about this Dijkstra-based algorithm is given in [13]. As shown in [13], the time and space complexities are $O((n \log n + m)\alpha(T))$ and $O((n + m)\alpha(T))$, respectively, where $\alpha(T)$ is the cost required for each function (defined in interval T) operation.

4. The Novel Voronoi-Based Index

We propose a novel voronoi-based index for kNN query over time-dependent road networks. In static road networks, the voronoi diagram divides the network (or space) into a group of disjoint subgraphs (or subspaces) where the nearest object of any vertex inside a subgraph is the object generating this subgraph. However, in time-dependent road networks, the nearest object of a vertex may be dynamic. The nearest object of a vertex v may be o_i for departure time $t \in [t_1, t_2]$ but it may be o_j for $t \in [t_3, t_4]$. The main idea of our novel voronoi-based index is also to divide the vertex set V into some vertex subsets V_i and every subset V_i is associated with one object $o_i \in O$. Different from static road networks, our voronoi-based index is time-dependent; that is, every vertex v inside a subset is with a time interval indicating when the object o_i is nearest to v. Next, we describe what the novel voronoi-based index is and how to construct it.

4.1. What Is the Voronoi-Based Index? Given a vertex v and an object o_i, $I_i(v)$ is called v's *maximum time interval* about o_i if it satisfies the following two conditions: (1) o_i is the nearest object of v for any departure time $t \in I_i(v)$ and (2) there does not exist another $I'_i(v) \supset I_i(v)$ satisfying the condition (1). Note that $I_i(v)$ may not be a continuous time interval; that is, if o_i is nearest to v for two disjoint departure time intervals $[t_1, t_2]$ and $[t_3, t_4]$, then $[t_1, t_2] \cup [t_3, t_4] \subseteq I_i(v)$. The voronoi-based index maintains a set C_i for every object $o_i \in O$, where C_i is a set of the tuples $(v, I_i(v))$ for all the vertices v with nonempty $I_i(v)$, i.e.,

$$C_i = \left\{\left(v, I_i(v)\right) \mid v \in V \wedge I_i(v) \neq \emptyset\right\} \quad (12)$$

We call C_i the *closest vertex-time pair set* (or closest pair set shortly) of o_i. For simplicity, we say v is a vertex in C_i if $(v, I_i(v)) \in C_i$. Next, we give the definition of the border vertex.

Definition 6 (border vertex). A vertex v_x in C_i is called a *border vertex* of C_i if there exist $v_y \in N^+(v_x)$ such that $(v_y, I_y) \notin C_i$ for any $I_y \supseteq f_{x,y}(I_i(v_x))$, where $N^+(v_x)$ is the outgoing neighbor set of v_x and $f_{x,y}(I_i(v_x))$ is the time interval mapped from $I_i(v_x)$ by the function $f_{x,y}(t) = t + w^*_{x,y}(t)$.

The border vertex v_x of C_i indicates that there exists a time point $t \in f_{x,y}(I_i(v_x))$ such that o_i is not the nearest object of v_y if one departs at time point t.

We use B_i to denote the set of all the border vertices of C_i. For every C_i, D_i is the set of mtt-functions $m_{x,n+i}(t)$ for all vertices v_x in C_i, that is,

$$D_i = \left\{m_{x,n+i}(t) \mid v_x \text{ is a vertex in } C_i\right\}, \quad (13)$$

and M_i is a matrix of size $|C_i| \times |B_i|$ to maintain mtt-function $m_{x,y}(t)$ for all pairs of vertex v_x and border vertex v_y in C_i, i.e.,

$$M_i = \left\{m_{x,y}(t) \mid v_x \in C_i \wedge v_y \in B_i\right\} \quad (14)$$

The voronoi-based index is $\{\mathsf{C}, \mathsf{B}, \mathsf{D}, \mathsf{M}\}$, where C, B, D, and M are the collections of all C_i, B_i, D_i, and M_i, respectively.

4.2. How to Construct The Voronoi-Based Index? We have explained how to compute mtt-function in Section 3. Next, we describe how to compute C_i and B_i for every $o_i \in O$.

For every vertex $v_x \in V$, $I_i(v_x)$ is initialized as the whole time domain T. We refine $I_i(v_x)$ iteratively by removing the subintervals on which $m_{x,n+i}(t)$ is larger than $m_{x,n+j}(t)$ for another object o_j. It means o_i is not the nearest object of v_x when departure time is in these subintervals. For every $o_j \in O$ ($o_j \neq o_i$), let $T_j(v_x)$ denote the maximum time interval on which $m_{x,n+j}(t) < m_{x,n+i}(t)$ and $I_i(v_x)$ is updated as $I_i(v_x) - T_j(v_x)$. After removing $T_j(v_x)$ for every other object o_j, if $I_i(v_x)$ is not empty, then the pair $(v_x, I_i(v_x))$ is inserted into C_i.

For every vertex v_x in C_i, if there exists an outgoing neighbor v_y of v_x, such that v_y is not in C_i or $f_{x,y}(I_i(v_x)) \not\subseteq I_i(v_y)$, then v_x must be a border vertex of C_i and it is inserted into B_i.

4.3. Query Processing by Voronoi-Based Index. Algorithm 2 describes how to find the k nearest objects for a query vertex v_q with departure time t_d. In Algorithm 2, $O(v_q)$ is a set to

Input: time-dependent graph G_T^*, query vertex v_q, departure time t_d and k
Output: the k nearest neighbor set $O(v_q)$
1: $O(v_q) \longleftarrow \emptyset, Q \longleftarrow \{C_q\}; E_q \longleftarrow \{v_q\}$
2: *while* $|O(v_q)| < k$ *do*
3: $C_i \longleftarrow$ DEQUEUE $(Q), O(v_q) \longleftarrow O(v_q) \cup \{o_i\}$;
4: *foreach* $v_y \in B_i$ *do*
5: *foreach* $v_x \in E_i$ *do*
6: $m_{q,y} \longleftarrow \min\{m_{q,y}, m_{q,x} + m_{x,y}(t_d + m_{q,x})\}$;
7: *foreach* $v_z \in N^+(v_y)$ *do*
8: *if* $m_{q,z} > m_{q,y} + w^*_{y,z}(t_d + m_{q,y})$ *then*
9: $m_{q,z} \longleftarrow m_{q,y} + w^*_{y,z}(t_d + m_{q,y})$;
10: Let C_j be the set including v_z when $t = t_d + m_{q,z}$;
11: *if* $C_j \notin O(v_q)$ *then*
12: $E_j \longleftarrow E_j \cup \{v_z\}$;
13: *if* $m_{q,n+j} > m_{q,z} + m_{z,n+j}(t_d + m_{q,z})$ *then*
14: $m_{q,n+j} \longleftarrow m_{q,z} + m_{z,n+j}(t_d + m_{q,z})$;
15: *if* $C_j \notin Q$ *then*
16: ENQUEUE(Q, C_j);
17: *else*
18: UPDATE(Q, C_j);
19: *return* $O(v_q)$

ALGORITHM 2: kNN-QUERY (G_T^*, v_q, t_d, k).

maintain the objects that have been found so far and Q is a priority queue to maintain a candidate set of C_i whose o_i is possible to be an object in kNN set. All $C_i \in Q$ are sorted in an ascending order by the minimum travel time $m_{q,n+i}$ from v_q to o_i. The top C_i in Q is with the minimum $m_{q,n+i}$ and it can be easily done using Fibonacci Heap. $O(v_q)$ and Q are initialized as $\emptyset$ and $\{C_q\}$, respectively, where C_q contains v_q for the departure time t_d, i.e., $(v_q, I_q(v_q)) \in C_q$ and $t_d \in I_q(v_q)$. $O(v_q)$ is expanded iteratively by inserting objects one by one from Q until $|O(v_q)| = k$. In each iteration, if $|O(v_q)| < k$, Algorithm 2 first dequeues the top C_i from Q with the minimum $m_{q,n+i}$. The object o_i of C_i must be one of k nearest objects of v_q. It can be guaranteed by Theorem 8. Then o_i will be inserted into $O(v_q)$. For every border vertex v_y in C_i, Algorithm 2 computes $m_{q,y}$ as $\min\{m_{q,x} + m_{x,y}(t_d + m_{q,x}) \mid v_x \in E_i\}$, where E_i is the entry set of C_i. The "entry" means any path entering into C_i must go through a vertex in E_i. E_i will be updated when Algorithm 2 runs. For every $v_z \in N^+(v_y)$, if $m_{q,z} > m_{q,y} + w^*_{y,z}(t_d + m_{q,y})$, then $m_{q,z}$ will be updated as $m_{q,y} + w^*_{y,z}(t_d + m_{q,y})$. Next, if v_z is in C_j ($C_j \neq C_i$ and $C_j \notin O(v_q)$) at the time point $t_d + m_{q,z}$, then v_z will be inserted into E_j as an entry of C_j. For the object o_j of C_j, $m_{q,n+j}$ will be updated as $m_{q,z} + m_{z,n+j}(t_d + m_{q,z})$ when $m_{q,n+j} > m_{q,z} + m_{z,n+j}(t_d + m_{q,z})$. If C_j is not in Q, then C_j will be enqueued into Q. Otherwise, C_j has been in Q and Q will be updated by C_j with new $m_{q,n+j}$. Algorithm 2 terminates when the size of $O(v_q)$ is k.

Example 7. We use the example in Figure 2 to illustrate the kNN querying process for $k = 3$. In this example, v_q is the query vertex and it is in C_1 for the departure time t_d. Q and $O(v_q)$ are initialized as $\{C_1\}$ and $\emptyset$, respectively. In the first iteration, C_1 is dequeued from Q and then o_1 is inserted into $O(v_q)$. Because v_1 is a border vertex of C_1 and v_3 is an outgoing neighbor of v_1, Algorithm 2 computes $m_{q,1}(t_d)$ and $m_{q,3}(t_d) = m_{q,1}(t_d) + w^*_{1,3}(t_d + m_{q,1}(t_d))$. Note that v_3 is in C_2 when $t = t_d + m_{q,3}(t_d)$ and then it is an entry of C_2. Therefore, C_2 is enqueued into Q. Similarly, C_3 is also enqueued into Q and $Q = \{C_2, C_3\}$. Assume that o_2 is nearer to v_q than o_3 and, in the second iteration, C_2 is dequeued and $O(v_q)$ is updated as $\{o_1, o_2\}$. In the same way, C_4 will be enqueued into Q in this iteration. In the final iteration, C_3 will be dequeued due to o_3 being nearer to v_q and then $O(v_q) = \{o_1, o_2, o_3\}$. Because $|O(v_q)| = 3$, Algorithm 2 terminates and returns $O(v_q)$.

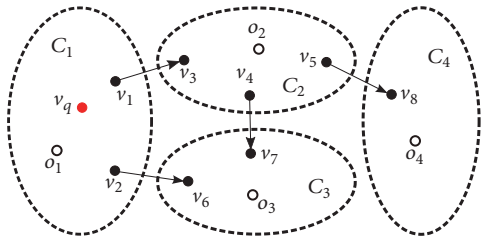


FIGURE 2: Query processing.

The next theorem guarantees the correctness of Algorithm 2.

Theorem 8. *In Algorithm 2, the object o_i of C_i dequeued from Q in the k-th iteration must be the k-th nearest object of query vertex v_q for the departure time t_d.*

Proof. We prove it by induction on k.

Basis. Obviously, C_q is dequeued from Q in the first iteration. By the definition of C_q, o_q is the nearest object of v_q when the departure time is t_d.

Induction. Assume that the i-th nearest neighbor of v_q is dequeued from Q in the i-th iteration for $i < k$. We need to prove it also holds for $i = k$. We prove it by contradiction. Let C_k be the closest pair set dequeued from Q in the k-th iteration and o_k is the object of C_k. Suppose that the k-th nearest object of v_q is $o_{k'}$ and $o_{k'} \neq o_k$. Let p be the shortest path from v_q to $o_{k'}$ with the departure time t_d. Because $k > 1$, then $C_{k'}$ is not C_q and there must exist an entry v_e of $C_{k'}$ in p. Letting v_b be the predecessor of v_e in p, then v_b must be a border vertex of C_b at time points $t_d + m_{q,b}$ and $C_b \neq C_{k'}$. There are two cases for the object o_b of C_b: (1) o_b is not in the k nearest object set of v_q and (2) o_b is in the k nearest object set of v_q.

For case (1), by the definition of C_b, o_b is the nearest neighbor of v_b at time point $t_d + m_{q,b}$, and then we have

$$m_{q,b} + m_{b,n+b}\left(t_d + m_{q,b}\right) < m_{q,b} + m_{b,n+k'}\left(t_d + m_{q,b}\right) \quad (15)$$

Thus o_b is nearer to v_q than $o_{k'}$ when the departure time is t_d. It means o_b must be in the k nearest object set of v_q, which is a contradiction.

For case (2), letting o_b be the i-th ($i < k$) nearest object of v_q, by the inductive assumption, C_b is dequeued from Q in i-th iteration. According to Algorithm 2, $C_{k'}$ is enqueued into Q in this iteration. Therefore, $C_{k'}$ will be dequeued from Q in k-th iteration instead of C_k, which is a contradiction. The proof is completed. □

Time and Space Complexities. The time and space complexities of Algorithm 2 are given below. Let b and e be the average size of B_i and E_i, respectively. In every iteration, Algorithm 2 upadates $m_{q,y}$ as $\min\{m_{q,y}, m_{q,x} + m_{x,y}(t_d + m_{q,x})\}$ for every border vertex v_y in C_i. It will cost $O(be)$ time. For every outgoing neighbor v_z of border vertex v_y, Algorithm 2 needs to compute $m_{q,z}$ and then it will cost $O(bd)$ time, where d is the average outdegree of the vertices in G_T. Therefore, the time complexity of Algorithm 2 is $O(kb(d+e))$. On the other hand, because Algorithm 2 needs to maintain $m_{q,y}$ and $m_{q,z}$, then the space complexity is $O(k(b+e))$.

5. Optimization of Voronoi-Based Index

In Section 4, we introduce the voronoi-based index on time-dependent graphs and propose Algorithm 2 to find the k nearest objects with voronoi-based index. Algorithm 2 dequeues the objects iteratively until all k nearest objects have been searched. In each iteration, Algorithm 2 uses a prior queue Q to maintain a candidate set of C_j whose object o_j may be one of the k nearest objects. This candidate set is essentially the set of all the closest pair sets adjacent to all the objects which have been searched in $O(v_q)$. However, for some closest pair set C_j of o_j, even though o_j are not one of k nearest objects, Algorithm 2 also needs to maintain C_j in Q and update the traveling time to o_j and the border vertices in C_j because C_j is adjacent to an object o_i in $O(v_q)$. It makes our algorithm not efficient enough for k nearest object query.

In this section, we propose a novel balanced tree structure on voronoi-based index, named V-tree, to organize all the closest pair sets C_j. V-tree can be considered as a secondary level index for k nearest object query. By V-tree, our algorithm can avoid maintaining the closest pair sets C_j whose objects o_j are not in the set of k nearest objects. The main idea of V-tree comes from G-tree, which is a balanced tree structure, proposed in [14]. G-tree cannot be used for k nearest object query on time-dependent graphs because of the following reasons:

(1) G-tree is built on the static graphs where the weight of every edge is a constant value. However, for time-dependent graphs, the weight on every edge is a time function $w_{i,j}(t)$ which is related to the departure time t. The value of $w_{i,j}(t)$ always changes with the change of departure time. Therefore, it is unfeasible to construct G-tree for time-dependent graphs.

(2) G-tree organizes all the vertices in a tree shape based on the distance of two vertices in a static graphs. Different from G-tree, V-tree is a second-level index based on voronoi-based index. It organizes all the closest pair sets C_i in a tree shape. Therefore, we need to define the distance between two closest pair sets C_i and C_j and then V-tree can be constructed.

(3) Because of above two points, the querying algorithm by G-tree cannot work for our problem; then we propose a novel querying algorithm by V-tree.

In the following, we first introduce what is V-tree and how to construct it based on voronoi-based index. Then we propose a novel querying algorithm for k nearest object query by V-tree. Finally, we utilize an example to explain why V-tree is more efficient than voronoi-based index.

5.1. What Is V-Tree. V-tree is a balanced tree structure to organize all the closest pair sets C_i in voronoi-based index. Specifically, every node in V-tree is a collection of some closest pair sets. We first give the definition of λ-partition for voronoi-based index.

Definition 9 (λ-partition for voronoi-based index). Given a collection C of some closest pair sets, a λ-partition of C is a set $\{\mathsf{C}_1 \cdots \mathsf{C}_\lambda\}$ satisfying three following conditions: (1) $\forall \mathsf{C}_i, \mathsf{C}_i \subset \mathsf{C}$; (2) $\forall \mathsf{C}_i, \mathsf{C}_j$ $(i \neq j), \mathsf{C}_i \cap \mathsf{C}_j = \emptyset$; (3) $\mathsf{C} = \bigcup_{1 \leq i \leq \lambda} \mathsf{C}_i$.

Note that every C_i is a subset of C and it is also a collection of closest pair sets. Every node in V-tree represents a collection C and has λ children by λ-partition on C; i.e., every C_i ($1 \leq i \leq \lambda$) is a child of C. Next, we introduce how to get an appropriate λ-partition for a given closest pair set C. An edge (v_x, v_y) is called a *crossing edge* from the closest pair sets C_i to C_j if v_x and v_y are the border vertices of C_i and C_j, respectively, i.e., $v_x \in B_i$ and $v_y \in B_j$. As our discussion in Section 2, the time function of a crossing edge is modeled

as a piecewise linear function $w_{x,y}(t)$. We define the *expected traveling time* $\overline{w}_{i,j}$ for a crossing edge (v_x, v_y) by the following:

$$\overline{w}_{x,y} = \frac{1}{T}\int_T w_{x,y}(t)\,dt \tag{16}$$

where T is the whole time interval for time function $w_{x,y}(t)$, i.e., $T = t_p - t_0$. Because $w_{x,y}(t)$ is a piecewise linear function, $\overline{w}_{x,y}$ can be calculated as follows:

$$\begin{aligned}\overline{w}_{x,y} &= \frac{1}{T}\int_T w_{x,y}(t)\,dt = \frac{1}{t_p - t_0}\int_{t_0}^{t_p} w_{x,y}(t)\,dt \\ &= \frac{1}{t_p - t_0}\sum_{i=1}^{p}\int_{t_{i-1}}^{t_i} w_i(t)\,dt \\ &= \sum_{i=1}^{p}\left(\frac{t_i - t_{i-1}}{t_p - t_0}\times\frac{w_i(t_{i-1}) + w_{i+1}(t_i)}{2}\right)\end{aligned} \tag{17}$$

Note that the expected traveling time $\overline{w}_{i,j}$ is a constant value to estimate the traveling time for crossing edge (v_x, v_y). Given two closest pair set C_i and C_j, we use $E_{i,j}$ to denote set of all the crossing edges from C_i to C_j, i.e., $E_{i,j} = \{(v_x, v_y) \mid v_x \in B_i \wedge v_y \in B_j\}$. Every closest pair set C_i can be regarded as a super vertex and there is an edge from C_i to C_j if $E_{i,j} \neq \emptyset$. The weight of such edge, denoted as $a_{i,j}$, is defined by the following:

$$a_{i,j} = \frac{1}{\left|E_{i,j}\right|}\sum_{(v_x,v_y)\in E_{i,j}} \overline{w}_{x,y} \tag{18}$$

It is worth noting that $a_{i,j}$ is the average value of the expected traveling time for all the crossing edge from C_i to C_j. $a_{i,j}$ is called the weight from C_i to C_j. Therefore, the λ-partition $\{\mathsf{C}_1, \ldots, \mathsf{C}_\lambda\}$ for a given collection C of some closest pair sets can be converted into a graph partition problem. For a pair of (C_i, C_j) with $E_{i,j} \neq \emptyset$, if C_i and C_j are in the same collection C_x $(1 \leq x \leq \lambda)$, then it is called an *inner edge* of C_x. Otherwise, C_i and C_j are in two different collections C_x and C_y, respectively, and then it is called an *interedge* from C_x to C_y. Correspondingly, the weight $a_{i,j}$ is called *inner weight* or *interweight,* respectively.

In V-tree, a parent node C is partitioned into λ children nodes $\{\mathsf{C}_1, \ldots, \mathsf{C}_\lambda\}$. To facilitate k nearest object query, an appropriate partition is expected such that the inner weights are less and the interweights are more. Thus the optimal λ-partition is to find a group of $\{\mathsf{C}_1, \ldots, \mathsf{C}_\lambda\}$ that minimizes the following objective function:

$$f(\{\mathsf{C}_1, \ldots, \mathsf{C}_\lambda\}) = \sum_{\mathsf{C}_x} a(\mathsf{C}_x) \tag{19}$$

where $a(\mathsf{C}_x) = \sum_{a_{i,j}\in\mathsf{C}_x} a_{i,j}$ is the summation of all the inner weights $a_{i,j}$ in C_x and $f(\{\mathsf{C}_1, \ldots, \mathsf{C}_\lambda\})$ is the summation of $a(\mathsf{C}_x)$ for all C_x. The optimal λ-partition on C is similar to the traditional graph partition problem which has been well studied [15–17]. However, the objective function is to minimize the sum of the interweights for the traditional graph partition. Therefore, for any pair (C_i, C_j) in C, if $E_{i,j} \neq \emptyset$, we assign a new weight $b_{i,j} = a_{\max} - a_{i,j}$ on it, where $a_{\max}$ represents the maximum weight for all the inner edges in C. Consider the following objective funtion:

$$g(\{\mathsf{C}_1, \ldots, \mathsf{C}_\lambda\}) = \sum_{\mathsf{C}_x,\mathsf{C}_y} b(\mathsf{C}_x, \mathsf{C}_y) \tag{20}$$

where $b(\mathsf{C}_x, \mathsf{C}_y) = \sum_{C_i\in\mathsf{C}_x, C_j\in\mathsf{C}_y} b_{i,j}$ is the summation of $b_{i,j}$ for all the inter edges from C_x to C_y and $g(\{\mathsf{C}_1, \ldots, \mathsf{C}_\lambda\})$ is the summation of $b(\mathsf{C}_x, \mathsf{C}_y)$ for all the pairs of C_x and C_y. The following theorem guarantees that the optimal λ-partition under objective functions $f(\cdot)$ and $g(\cdot)$ is the same.

Theorem 10. *The λ-partition minimizing $f(\cdot)$ is exactly the λ-partition minimizing $g(\cdot)$.*

Proof. Given a λ-partition $\{\mathsf{C}_1, \ldots, \mathsf{C}_\lambda\}$ on C, let $a(\mathsf{C}_x, \mathsf{C}_y)$ denote the summation of all the inter weights $a_{i,j}$ from C_x to C_y, that is, $a(\mathsf{C}_x, \mathsf{C}_y) = \sum_{C_i\in\mathsf{C}_x, C_j\in\mathsf{C}_y} a_{i,j}$, and then we have

$$a(\mathsf{C}) = \sum_{\mathsf{C}_x,\mathsf{C}_y} a(\mathsf{C}_x, \mathsf{C}_y) + \sum_{\mathsf{C}_x} a(\mathsf{C}_x) \tag{21}$$

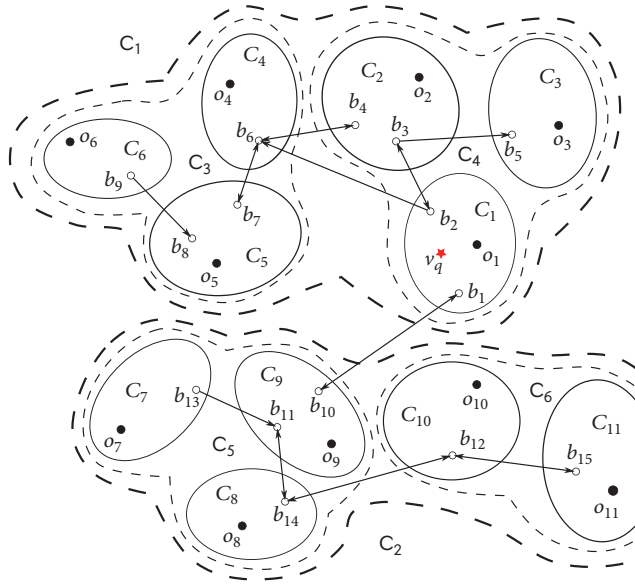
Because $a(\mathsf{C})$ is independent of λ-partition on C, then minimizing $\sum_{\mathsf{C}_x} a(\mathsf{C}_x)$ is equivalent to maximizing $\sum_{\mathsf{C}_x,\mathsf{C}_y} a(\mathsf{C}_x, \mathsf{C}_y)$. On the other hand, because $b_{i,j} = a_{\max} - a_{i,j}$ and $b_{i,j} \geq 0$, then maximizing $\sum_{\mathsf{C}_x,\mathsf{C}_y} a(\mathsf{C}_x, \mathsf{C}_y)$ is equivalent to minimizing $\sum_{\mathsf{C}_x,\mathsf{C}_y} b(\mathsf{C}_x, \mathsf{C}_y)$. Therefore, the λ-partition minimizing the objective function $f(\cdot)$ also minimizes the objective function $g(\cdot)$. □

Thus the optimal λ-partition under objective function $g(\cdot)$ is a traditional graph partition problem. In this paper, we adopt the multilevel graph partitioning technique (METIS) proposed by Metis et al. in [15], which is a classic graph partition algorithm.

Next, we introduce the concept of the *minimum distance*, which is to be maintained in V-tree. For two closest pair set C_i and C_j, if $E_{i,j} \neq \emptyset$, we defined the *minimum weight* from C_i to C_j, denoted as $w_{i,j}^{\min}$, by the following:

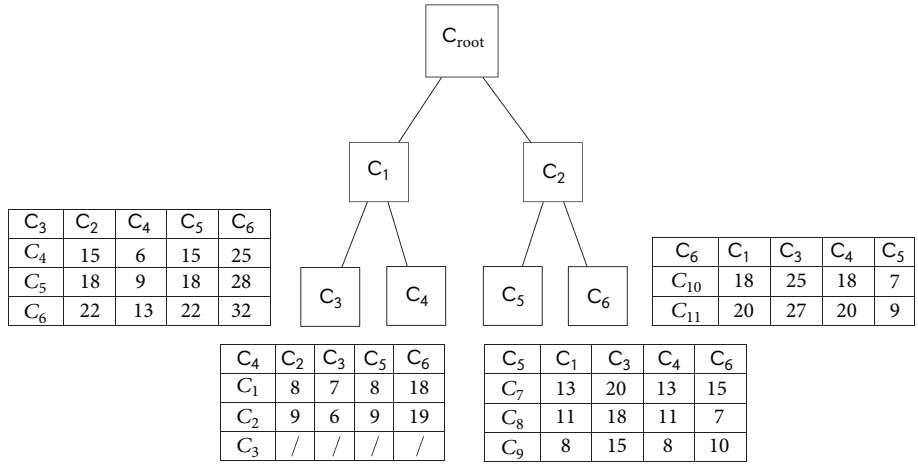
$$w_{i,j}^{\min} = \min_{(v_x,v_y)\in E_{i,j}}\left\{\min_t\left\{w_{x,y}(t)\right\}\right\} \tag{22}$$

where $(v_x, v_y) \in E_{i,j}$ is a crossing edge from C_i to C_j and $\min_t\{w_{x,y}(t)\}$ is the minimum value of time funtion $w_{x,y}(t)$. Thus $w_{i,j}^{\min}$ is the minimum value of $\min_t\{w_{x,y}(t)\}$ for all the crossing edge in $E_{i,j}$. A new graph can be constructed from the voronoi-based index; every closest pair set C_i can be regarded as a super vertex. If $E_{i,j} \neq \emptyset$, there is an edge from C_i to C_j and its weight is $w_{i,j}^{\min}$. Therefore, for any two C_i and C_j, the shortest distance from C_i to C_j in this graph is called the *minimum distance* from C_i to C_j and we denote it by $d_{i,j}$. Especially, for a closest pair set C_i and a collection C, the minimum distance from C_i to C is denoted by $d(C_i, \mathsf{C})$ and is the minimum value of $d_{i,j}$ for all C_j in C, i.e., $d(C_i, \mathsf{C}) = \min_{C_j\in\mathsf{C}}\{d_{i,j}\}$. The value of $d(C_i, \mathsf{C})$ can be calculated easily by existing single-source shortest distance algorithm. Note that



$m_{q,n+1}$	1
$m_{q,n+2}$	3
$m_{q,n+3}$	5
$m_{q,n+4}$	8
$m_{q,n+5}$	12
$m_{q,n+6}$	/
$m_{q,n+7}$	/
$m_{q,n+8}$	14
$m_{q,n+9}$	10
$m_{q,n+10}$	19
$m_{q,n+11}$	22

Figure 3: A 2-Partition on a voronoi-based index.



C_3	C_2	C_4	C_5	C_6
C_4	15	6	15	25
C_5	18	9	18	28
C_6	22	13	22	32

C_6	C_1	C_3	C_4	C_5
C_{10}	18	25	18	7
C_{11}	20	27	20	9

C_4	C_2	C_3	C_5	C_6
C_1	8	7	8	18
C_2	9	6	9	19
C_3	/	/	/	/

C_5	C_1	C_3	C_4	C_6
C_7	13	20	13	15
C_8	11	18	11	7
C_9	8	15	8	10

Figure 4: V-tree of the voronoi-based index in Figure 3.

$d(C_i, \mathsf{C}) = 0$ when $C_i \in \mathsf{C}$. Next, we give the definition of V-tree below.

Definition 11 (V-tree). A V-tree, denoted as $\mathcal{T}_V$, is a balanced tree $\mathcal{T}_V$ that satisfies the following conditions:

(1) Every node in V-tree represents a collection C of some closest pair sets. Specially, the root node represents the collection C_{root} of all the closest pair sets in voronoi-based index;

(2) Every nonleaf node C has λ children $\{\mathsf{C}_1, \ldots, \mathsf{C}_\lambda\}$. All these children consist of the optimal λ-partition on C.

(3) Every leaf node C has at most σ closest pair sets.

In V-tree, every leaf node C is associated with a matrix to maintain the minimum distance from every closest pair set C_i in leaf node C to every node C′ in V-tree $\mathcal{T}_V$, i.e.,

$$\left\{d\left(C_i, \mathsf{C}'\right) \mid C_i \in \mathsf{C} \wedge \mathsf{C}' \in \mathcal{T}_V\right\}, \tag{23}$$

where C′ and C are different nodes in $\mathcal{T}_V$. It is because $C_i \in \mathsf{C}$ and then $d(C_i, \mathsf{C}) = 0$. Note that if C′ is an ancestor C in $\mathcal{T}_V$, we also have $d(C_i, \mathsf{C}') = 0$ directly.

Example 12. Figure 3 illustrates an example of 2-partition on a voronoi-based index. The corresponding V-tree is shown in Figure 4. In this example, C_{root} is the collection of all the closest pair set and it is partitioned into $\{\mathsf{C}_1, \mathsf{C}_2\}$. Thus C_1 and C_2 are two children of C_{root} in V-tree. Similarly, C_3 and C_4 are two children of C_1. C_5 and C_6 are two children of C_2. Because C_3, C_4, C_5, and C_6 are four leaf nodes of V-tree, then

Input: a collection C_{root} of all the closest pair sets in voronoi-based index, λ, σ
Output: V-tree $\mathcal{T}_V$
1: $Q \longleftarrow \{\mathsf{C}_{root}\}$, $\mathcal{T}_V \longleftarrow \{\mathsf{C}_{root}\}$; /* V-tree $\mathcal{T}_V$ only have one node C as its root */
2: *while* $Q \neq \emptyset$ *do*
3: C $\longleftarrow$ Dequeue (Q);
4: *if* |C| > σ *then*
5: Children(C) $\longleftarrow$Partition (C);
6: inserts Children(C) into $\mathcal{T}_V$ as the children of C;
7: *for* each $\mathsf{C}_x \in$ Children(C) *do*
8: Enqueue(Q, C_x);
9: *for* each leaf node C in $\mathcal{T}_V$ *do*
10: calculates $\{d(C_i, \mathsf{C}') \mid C_i \in \mathsf{C} \wedge \mathsf{C}' \in \mathcal{T}_v\}$ and maintains it on C;
11: *return* $\mathcal{T}_V$

Algorithm 3: Construct-Vtree ($\mathsf{C}_{root}, \lambda, \sigma$).

every one of them is associated with a minimum distance matrix. For example, C_1 is a closest pair set in C_4 and the minimum distance from C_1 to every node C_i in $\mathcal{T}_V$ is stored in the matrix associated with C_4. In Figure 4, every matrix maintaining the minimum distance is presented next to every node in V-tree.

5.2. How to Construct V-Tree. Algorithm 3 describes how to constrcut a V-tree $\mathcal{T}_V$ for a given collection C of all the closest pair sets in a voronoi-based index. To construct V-tree $\mathcal{T}_V$, Algorithm 3 utilizes a queue Q to dequeue the nodes in $\mathcal{T}_V$ iteratively and expand the children for every node until V-tree has been constructed. Both Q and $\mathcal{T}_V$ are initialized as $\{\mathsf{C}_{root}\}$, where C_{root} is the collection of all the closest pair set in voronoi-based index. In each iteration, a node C is dequeued from Q. If $|\mathsf{C}| > \sigma$, C is a nonleaf node and then Algorithm 3 calls function Partition (C) to get the optimal λ-partition on C. Note that |C| is the number of the closest pair sets in C and Partition (C) is the implementation of METIS algorithm. Assume that the output of Partition (C) is $\{\mathsf{C}_1, \ldots, \mathsf{C}_\lambda\}$, which will be inserted into $\mathcal{T}_V$ as the children set Children(C) of C. Next, Algorithm 3 enqueues every child C_x of C into Q. Note that if $|\mathsf{C}| \leq \sigma$, node C is a leaf node in $\mathcal{T}_V$ and Algorithm 3 will dequeue next top element in Q. When Q is empty, the structure of V-tree $\mathcal{T}_V$ has been constructed. Finally, for each leaf node C in $\mathcal{T}_V$, Algorithm 3 calculates the minimum distance matrix $\{d(C_i, \mathsf{C}') \mid C_i \in \mathsf{C} \wedge \mathsf{C}' \in \mathcal{T}_v\}$ and associates it with the leaf node C. Therefore, V-tree is constructed and returned.

Example 13. We use the example in Figure 3 to show how to construct V-tree in Figure 4. In this example, $\lambda = 2$ and $\sigma = 3$. At the beginning, Q and $\mathcal{T}_V$ are initialized as $\{\mathsf{C}_{root}\}$; C_{root} is the collection of all the closest pair set in Figure 3. In the first iteration, the root node C_{root} is dequeued from Q, because $|\mathsf{C}_{root}| = 11 \geq \sigma$ and C_{root} is partitioned into C_1 and C_2. Then C_1 and C_2 are inserted into $\mathcal{T}_v$ as two children of C_{root}. Next, C_1 and C_2 are enqueued into Q. In second and third iteration, C_1 and C_2 are dequeued from Q, respectively. $\{\mathsf{C}_3, \mathsf{C}_4\}$ and $\{\mathsf{C}_5, \mathsf{C}_6\}$ are inserted into $\mathcal{T}_V$ and enqueued into Q. Algorithm 3 dequeues C_3, C_4, C_5, and C_6 one by one. Finally, the minimum distance matrices on C_3, C_4, C_5, and C_6 are, respectively, computed and maintained in V-tree $\mathcal{T}_V$. Thus the V-tree shown in Figure 4 has been constructed.

Space Complexity of V-Tree. The space complexities of V-tree are given below. Let h denote the height of a V-tree $\mathcal{T}_V$. Obviously, $O(h) = O(\log_\lambda(|O|/\sigma) + 1)$, where $|O|$ is the number of objects in G_T. For the i-th level of $\mathcal{T}_V$, there are at most λ^i nodes. Thus the number of nodes in a V-tree $\mathcal{T}_V$ is

$$O\left(\frac{1-\lambda^{\log_\lambda(|O|/\sigma)+1}}{1-\lambda}\right) = O\left(\frac{\lambda\left(|O|/\sigma\right)-1}{\lambda-1}\right) = O\left(\frac{|O|}{\sigma}\right) \tag{24}$$

For each leaf node C in $\mathcal{T}_V$, a matrix is used to maintain the minimum distance from every closest pair set $C_i \in \mathsf{C}$ to every node $\mathsf{C}' \in \mathcal{T}_V$. The size of every matrix is $O(\sigma \times |O|/\sigma) = O(|O|)$. Because there are at most $O(|O|/\sigma)$ leaf nodes in $\mathcal{T}_V$, then the space complexity of $\mathcal{T}_V$ is $O(|O|^2/\sigma + |O|/\sigma) = O(|O|^2/\sigma)$.

For the *time complexity* of V-tree construction, Algorithm 3 needs to compute λ-partition using existing partition algorithm for every nonleaf nodes in $\mathcal{T}_V$. We use $O(P_\lambda)$ to denote the time complexity of λ-partition on all the closest pair sets in voronoi-based index. Therefore, constructing the structure of $\mathcal{T}_V$ needs $O(|O|/\sigma \times P_\lambda)$ time. For every leaf node C in $\mathcal{T}_V$, the minimum distance needs to be calculated. The minimum distances from one $C_i \in \mathsf{C}$ to all other C_j can be calculated by single-source and multidestination shortest path algorithm, which can be done in $O(|O|^2)$ time. Then the time complexity for every matrix on leaf node is $\sigma|O|^2$. Therefore, the time complexity of V-tree construction is $O(|O|/\sigma \times P_\lambda + |O|/\sigma \times \sigma|O|^2) = O(P_\lambda|O|/\sigma + |O|^3)$.

5.3. Query Processing with V-Tree. In this section, we propose an efficient algorithm for searching k nearest objects by V-tree. The pseudocode is shown in Algorithm 4. The same as

Input: query vertex v_q, departure time t_d and k
Output: the top-k nearest neighbor set $O(v_q)$
1: $Q \longleftarrow \emptyset$, $O(v_q) \longleftarrow \emptyset$, $R \longleftarrow \{\mathsf{C}_{root}\}$;
2: *while* $|O(v_q)| < k$ and $Q \neq \emptyset$ *do*
3: let C_i and C are the top elements in Q and R respectively;
4: *if* $m_{q,n+i} \leq d(C_q, \mathsf{C})$ *then*
5: $C_i \longleftarrow$ Dequeue (Q), $O(v_q) \longleftarrow O(v_q) \cup \{o_i\}$;
6: $B(O(v_q)) \longleftarrow B(O(v_q)) \cup B_i$;
7: *for* each $v_x \in B(O(v_q))$ *do*
8: *if* $v_x \in C_i$ or $v_x \in N^+(v_y) \wedge v_y \in C_i$ *then*
9: updates $m_{q,x}$;
10: *for* each $C_j \in Q$ *do*
11: *if* C_j has an entry vertex connected to a border vertex in $B(O(v_q))$ *then*
12: updates $m_{q,n+j}$;
13: *else*
14: $\mathsf{C} \longleftarrow$ Dequeue (R);
15: *if* C is a non-leaf node in $\mathscr{T}_V$ *then*
16: *for* each $\mathsf{C}_x \in$ Children(C) *do*
17: Enqueue(R, C_x);
18: *else*
19: *for* each $C_j \in \mathsf{C}$ *do*
20: Enqueue(Q, C_j);
21: *return* $O(v_q)$

Algorithm 4: kNN-Query (v_q, t_d, k).

Algorithm 2, $O(v_q)$ is a set of the objects that have been found up to now. Different from Algorithm 2, Algorithm 4 needs two priority queues Q and R, where Q is used to maintain some closest pair sets C_i whose o_i is possible to be one of k nearest objects and R is used to maintain some nodes C of V-tree. All C_i in Q are sorted in an ascending order by the minimum travel time $m_{q,n+i}$ from v_q to o_i. All C in R are sorted in an ascending order by the minimum distance $d(C_q, \mathsf{C})$. Initially, $O(v_q) \longleftarrow \emptyset$, $Q \longleftarrow \emptyset$, and $R \longleftarrow \{\mathsf{C}_{root}\}$. In each iteration, Algorithm 4 first considers the top elements C_i in Q and, C in R, there are two cases: (1) $m_{q,n+i} \leq d(C_q, \mathsf{C})$ and (2) $m_{q,n+i} > d(C_q, \mathsf{C})$.

For case (1), C_i is dequeued from Q with $m_{q,n+i}$; the object o_i of C_i must be one of k nearest objects of v_q. It can be guaranteed by Theorem 14 and then o_i will be inserted into $O(v_q)$. Let $B(O(v_q))$ denote the set of the border vertices for all C_i whose $o_i \in O(v_q)$, i.e., $B(O(v_q)) = \bigcup_{o_i \in O(v_q)} B_i$. Algorithm 4 first updates $B(O(v_q))$ by $B(O(v_q)) \cup B_i$ and then updates the value of $m_{q,x}$ for any $v_x \in B(O(v_q))$ if $v_x \in C_i$ or v_x is an outgoing neighbor of v_y, $v_y \in C_i$. Next, Algorithm 4 updates $m_{q,n+j}$ for any $C_j \in Q$ by the border vertices in $B(O(v_q))$, if C_j has an entry vertex connected to a border vertex in $B(O(v_q))$. The order of C_j in Q is also be updated when $m_{q,n+j}$ changes. The process for updating $m_{q,x}$ and $m_{q,n+j}$ is the same as that in Algorithm 2. Note that in Algorithm 2, when C_i is dequeued, all $m_{q,n+j}$ for its neighboring C_j are needed to be updated, but for Algorithm 4, only some neighboring $C_j \in Q$ are needed to be updated. Therefore, in each iteration, Algorithm 4 only needs to update $m_{q,n+j}$ for a small number of C_j in Q which are a subset of the closest pair sets to be updated in Algorithm 2.

For case (2), C is dequeued from R. If C is a nonleaf node in V-tree, the children C_x $(1 \leq x \leq \lambda)$ of C will be inserted into R with the minimum distance $d(C_q, \mathsf{C}_x)$. Note that $d(C_q, \mathsf{C}_x)$ is maintained in the leaf node C_q containing C_q in V-tree. If C is a leaf node in V-tree, every closest pair set $C_i \in \mathsf{C}$ will be enqueued into Q with $m_{q,n+i}$. Algorithm 4 calculates $m_{q,n+i}$ by the following:

$$\min_{v_x \in B(O(v_q)), v_y \in C_i, (v_x, v_y) \in E} \left\{ m_{q,x} + w^*_{x,y} \left(t_d + m_{q,x} \right) + m_{y,n+i} \right\} \tag{25}$$

Note that the process to calculate $m_{q,n+i}$ is as the same as that in Algorithm 2.

Algorithm 4 terminates when the size $O(v_q)$ is k. It means that k nearest objects of query vertex v_q have been found.

Theorem 14 guarantees the correctness of Algorithm 4.

Theorem 14. *In Algorithm 4, the object node o_i inserted into $O(v_q)$ at the k-th time must be the k-th nearest object of query vertex v_q for the departure time t_d.*

Proof. Similar to the proof of Theorem 8, we prove it by induction on k.

Basis. Obviously, the object node o_q in C_q is the first object node inserted into $O(v_q)$. By the definition of C_q and o_q, we know that o_q is indeed the nearest object of v_q for the departure time t_d.

Induction. Assume that the i-th nearest neighbor of v_q is inserted into the result set $O(v_q)$ at the i-th time for $i < k$.

We need to prove it also hold for $i = k$. We prove it by contradiction. It can be known that the object node inserted into $O(v_q)$ at the i-th time is the object node that dequeued from the priority queue Q at the i-th time. Let C_k be the closest pair set dequeued from Q at the k-th time. Suppose that the k-th nearest object of v_q is $o_{k'}$ and $o_{k'} \neq o_k$. Let p be the shortest path from v_q to $o_{k'}$ with the departure time t_d. Because $k > 1$, $C_{k'}$ is not C_q and there must exist an entry v_e of $C_{k'}$ in p. Let v_b be the predecessor of v_e in p; then v_b must be a border vertex of C_b at time points $t_d + m_{q,b}$ and $C_b \neq C_{k'}$. Obviously, for the object node o_b of C_b, (1) o_b is not in the k nearest object set of v_q and (2) o_b is in the k nearest object set of v_q.

For case (1), it is similar to this case in Theorem 8;

For case (2), it is assumed that o_b is the i-th ($i < k$) nearest object of v_q. According to Algorithm 4, we know that $B(O(v_q))$ will be updated, when o_b is inserted into $O(v_q)$. At this point, $m_{q,n+k'}$ will be updated by $B(O(v_q))$ if $C_{k'}$ is in Q. Otherwise, $m_{q,n+k'}$ will be updated by $B(O(v_q))$ when $C_{k'}$ is enqueued in Q. So $C_{k'}$ will be dequeued from Q at k-th time with an accurate $m_{q,n+k'}$, which is a contradiction. The proof is completed. □

Time and Space Complexities. We first analyze the time complexity of Algorithm 4. For processing priority queue Q, the time complexity is the same as Algorithm 2 in the worst case. For processing priority queue R, all nodes in $\mathcal{T}_V$ will be dequeued in the worst case; then the time complexity is $O(|O|/\sigma)$. Therefore, the time complexity of Algorithm 4 is $O(kb(d+e) + |O|/\sigma)$. Next, we analyze the space complexity, the same as Algorithm 2; the space complexity of Algorithm 4 is $O(k(b+e))$.

5.4. Effectiveness of V-Tree. We use the example in Figure 3 to show Algorithm 4 with V-tree being more efficient than Algorithm 2 without V-tree. For this example, we set $k = 3$, $\lambda = 2$, and $\sigma = 3$. The red five-pointed star represents the query vertex v_q which is in C_1 for the departure time t_d. The minimum distance $m_{q,n+i}$ from v_q to every object o_i is shown in the table in Figure 3.

For Algorithm 2, Q is initialized as $\{C_1\}$. In the first iteration, C_1 is dequeued from Q. Because C_2, C_4, and C_9 are three neighboring closest pair set of C_1, Algorithm 2 will update their $m_{q,n+i}$ and then enqueue them into Q. Similarly, in the second iteration, C_2 is dequeued from Q. C_3 and C_4 are enqueued into Q. C_3 will be dequeued from Q in the final iteration. Note that $m_{q,n+i}$ for the object in C_4 will be updated twice in the first and second iteration, respectively; this is because C_4 is the neighboring closest pair set of both C_1 and C_2. Therefore, Algorithm 2 needs to update $m_{q,n+i}$ for five times in the whole querying process.

For Algorithm 4 by V-tree, R and Q are initialized as $\{C_{root}\}$ and $\{\emptyset\}$, respectively. When C_4 is dequeued from R, C_1,C_2, and C_3 are enqueued into Q. Note that $B(O(v_q)) = \emptyset$; then Algorithm 4 does not need to update $m_{q,n+i}$ for C_1, C_2, and C_3. In the following iterations, Algorithm 4 will dequeue C_1, C_2, and C_3 from Q one by one. This is because for the top element C_3 in R, $d(C_q, C_3) = 7$ is always larger than $m_{q,n+i}$ for C_1, C_2, and C_3 which are the top elements in Q. When C_1 is dequeued from Q, $m_{q,n+2}$ is updated and C_2 is enqueued into Q. Similarly, $m_{q,n+3}$ will be updated and C_3 will be enqueued into Q. When C_3 is dequeued from Q, Algorithm 4 terminates and it only needs to update $m_{q,n+i}$ twice in the whole querying process, which is obviously more efficient than Algorithm 2.

Next, we give the following theorem to guarantee that the Algorithm 4 with V-tree is at least as good as Algorithm 2.

Theorem 15. *The number of updating $m_{q,n+i}$ for all the closest pair sets in Algorithm 4 is no more than that in Algorithm 2.*

Proof. In each iteration, for Algorithm 2, when C_j is dequeued, all $m_{q,n+i}$ for its neighboring C_i are needed to be updated. Obviously, for Algorithm 4, it only needs to update $m_{q,n+i}$ for a small number of C_i in Q which are a subset of the closest pair sets to be updated in Algorithm 2. So the number of updating $m_{q,n+i}$ for all the closest pair sets in Algorithm 4 is no more than that in Algorithm 2. □

6. Experiements

We compare our V-tree method (marked as VT) and voronoi-based index method (marked as VI) with FTTI (Fast-Travel-Time Index) method [11] and TLNI (Tight-and-Loose-Network Index) method [9] on the real-life datasets. FTTI and TLNI are the state of the art index-based methods for kNN query over time-dependent road networks. Note that FTTI and TLNI are used on G_T^* in which every edge is an nwt-function $w_{i,j}^*(t)$ because FTTI and TLNI do not allow the waiting time. Although some algorithms are proposed in recent works [5, 10], they are only to find the nearest object (i.e., $k = 1$) and they cannot be used for general kNN query on time-dependent graphs. All the experiments are conducted on a 2.6GHz Intel Core i7 CPU PC with the 16GB main memory, running on Windows 7.

6.1. DataSets and Experiment Setup. We tested the voronoi-based index method on California road network (CARN) (http://snap.stanford.edu/data/roadNet-CA.html) with 196,5206 vertices and 553,3214 edges. We extracted five time-dependent graphs with different size using the CARN dataset. The number of vertices ranges from 100k to 500k. The time domain is set as $T = [0, 2000]$; i.e., the departure time t can be selected from $[0, 2000]$ for any vertex. Here, 2000 means 2000 time units. For every $w_{i,j}(t)$, we split the time domain T to p subintervals and assign a linear function randomly for every subinterval and then $w_{i,j}(t)$ is a piecewise linear function.

6.2. Experienmental Results

Exp-1. Impact of Network Size. In this group of experiments, we study the impact of time-dependent network size. The number of the vertices increases from 100k to 500k and the number of objects is fixed at 10k. We investigate the querying time for $k = 7$. The number of piecewise intervals of $w_{i,j}(t)$ is set as 4. As shown in Figures 5(a) and 5(b), the querying

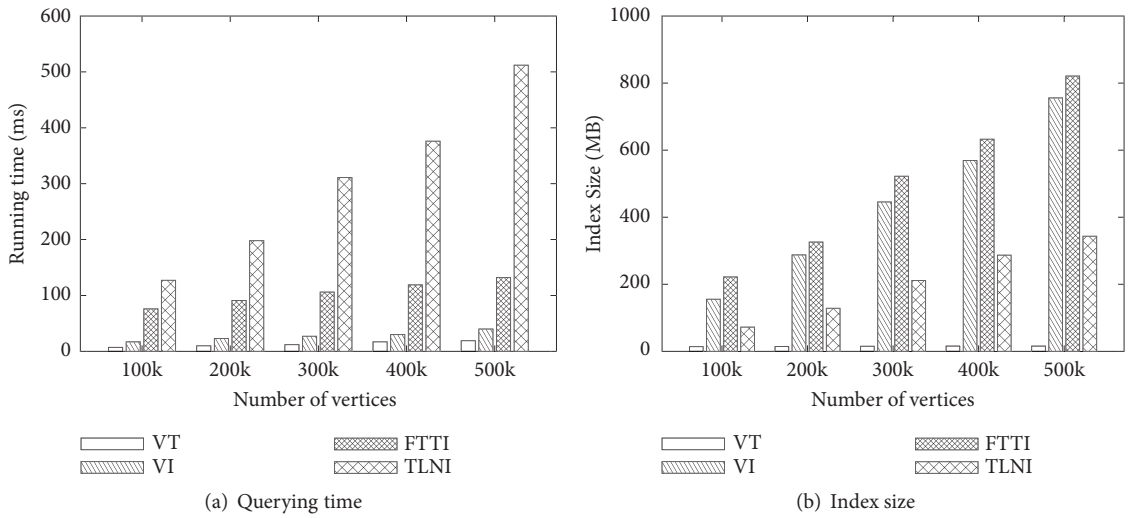


Figure 5: Impact of the network size.

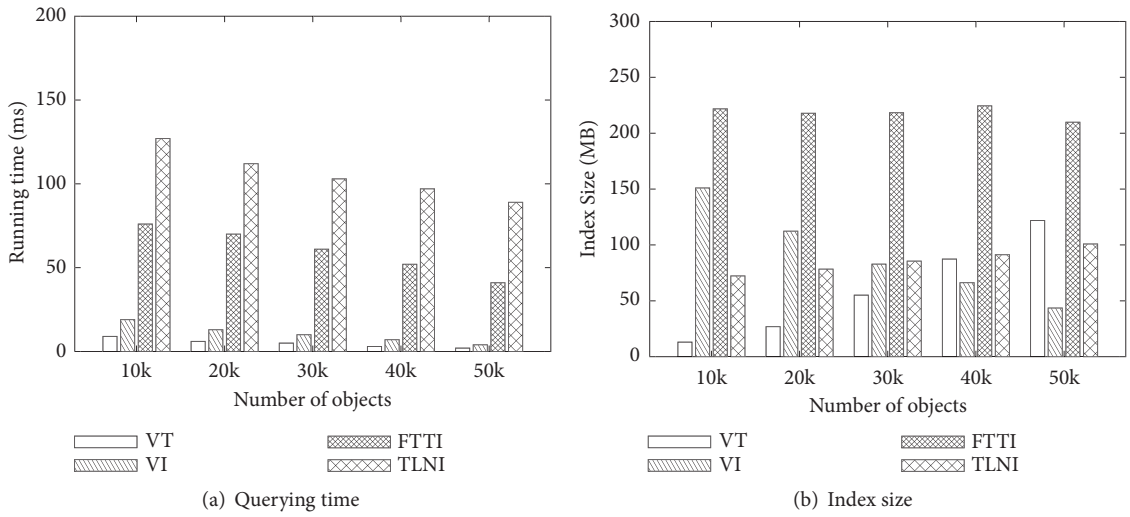
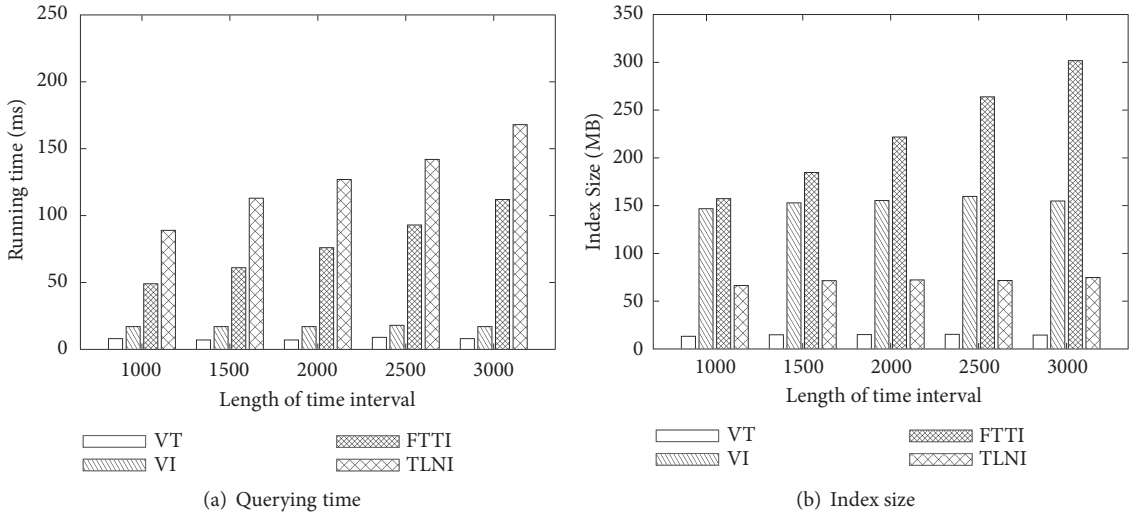


Figure 6: Impact of the object set size.

time of our method is always less than FTTI and TLNI. Note that our V-tree method has the minimum querying time. Specifically, the querying time of TLNI is always much more than our methods even though TLNI has the smallest index size. The reason TLNI index only maintains the vertices for an object o_i is that the upper bound of travel time to o_i is less than the lower bound to the other objects. It cannnot facilitate query effectively in large networks.

Exp-2. Impact of Object Set Size. In this group of experiments, the number of the vertices is fixed at 100k and the number of objects ranges from 10k to 50k. As shown in Figures 6(a) and 6(b), the querying time of our methods is always less than FTTI and TLNI. The querying time decreases with the increasing of the object set size. There are two reasons as follows: (1) the average size of C_i and B_i decreases if the object set size increases; (2) the increasing of object size results in that the objects become nearer to v_q and then querying time decreases. The index size for VI, FTTI, and TLNI decreases with increasing of the object size. But for V-tree, the index size increases with the object size increasing. This is because more objects result in more nodes in V-tree.

Exp-3. Impact of the Time Domain. In Figure 7, we study the impact of time domain. In this group of experiments, the number of vertices and objects is fixed at 100k and 10k, repectively. The time domain ranges from $[0, 1000]$ to $[0, 3000]$. We investigate the querying time for $k = 7$. As shown in Figures 7(a) and 7(b), the querying time and index size of our methods are not affected by the expanding of time domain. However, for FTTI and TLNI, the querying time increases with the the expanding of time domain. This is because they need to maintain the estimated value about travel time in index to facilitate kNN query. If the time



(a) Querying time (b) Index size

Figure 7: Impact of the length of time interval.

domain becomes larger, the deviation between the estimation and the actual travel time will become larger too. It cannot facilitate query effectively.

Exp-4. Impact of the Number of Piecewise Intervals. In Figure 8, we investigate the impact of the number of piecewise intervals of $w_{i,j}(t)$. In this group of experiments, the number of piecewise intervals of $w_{i,j}(t)$ increases from 2 to 10. The numbers of the vertices and objects are fixed at 100k and 10k, repectively. As shown in Figures 8(a) and 8(b), the querying time and index size always increase with the increasing of the number of piecewise intervals. The reason is that the more piecewise intervals of $w_{i,j}(t)$, the more piecewise intervals of mtt-function and then the more border vertices maintained in the index.

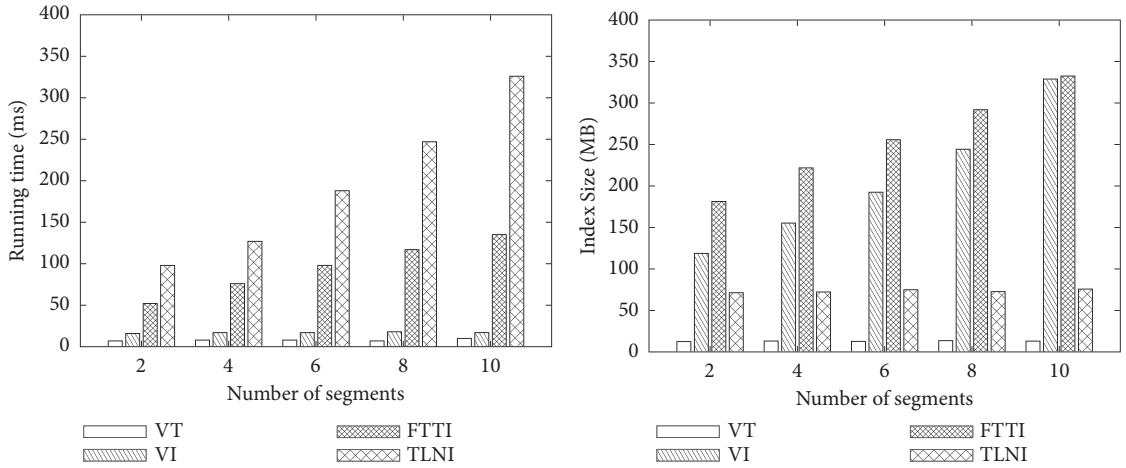
Exp-5. Impact of the Average Distance to Objects. The impact of the average distance to objects is shown in Figure 9. In this group of experiments, the number of vertices is set as 100k and 500k, respectively. The number of objects is fixed at 10k and $k = 7$. The average distance to objects ranges from 4 to 12. Note that the distance means the hops from the query vertex to an object. As shown in Figures 9(a) and 9(b), the querying time increases marginally with the increasing of distance. Our methods are always better than FTTI and TLNI.

Exp-6. Impact of k. In Figure 10, we study the querying time by varying k from 1 to 10 on two different networks with 10k vertices and 50k vertices, respectively. In this group of experiments, the number of objects is fixed at 10k and 50k for two different networks, respectively. As shown in Figures 10(a) and 10(b), the querying time always increases marginally with the increasing of k for our index method.

Exp-7. Impact of λ and σ. In Figure 11, we investigate the impact of the λ and σ for V-tree. In this group of experiments, the numbers of the vertices and objects are fixed at 100k and 10k, repectively. The values of λ and σ are selected from $[2, 4, 6, 8, 10]$ and $[10, 20, 30, 40]$, respectively. The experimental results shows the effectiveness of V-tree is the best when $\lambda = 4$. The querying time increases with the increasing of σ. But the index size increases with the decreasing of σ. The index size is not affected by λ varying.

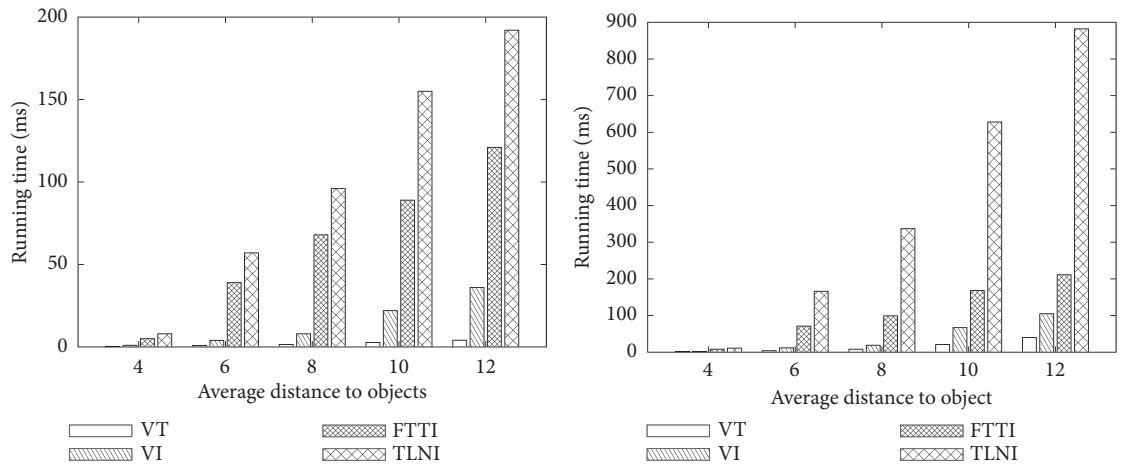
7. Related Work

kNN query has been well studied on static road networks. Most of the existing works propose various index techniques. The main ideas of these methods are to partition the vertices into several clusters, and then the clusters are organized as a voronoi diagram or a tree (e.g., R-tree) [14, 18–25]. These methods precompute and maintain the shortest distances for some pairs of vertices to facilitate kNN query. Unfortunately, these index techniques cannot be used for the time-dependent road networks because the minimum travel time between two vertices always varies with time. kNN query has also been studied on time-dependent road networks [9–12, 26]. Most of these works are based on A$*$ algorithm. The authors in [5, 10] study the problem to find nearest (i.e., $k = 1$) object on time-dependent networks. In [10], a virtual node v is inserted into the graph G with the zero-cost edges connecting to all the objects. The nearest object can be found on the shortest path from the query vertex to v. The authors in [27] study the problem of finding k POIs that minimize the aggregated travel time from a set of query points. The index-based methods are proposed in [9, 11]. In [9], a $*$ algorithm is utilized to expand the road networks by estimating an upper or lower bound of travel time. An index is built to facilitate kNN query using these estimated bounds. In [11], time domain is divided to several subintervals. For every subinterval, C nearest objects of every vertex are found by an estimation of minimum travel time. There are two main drawbacks of these methods. First, in these works, the FIFO (first in first out) property is required for networks and waiting time is not allowed. Second, the indexes proposed by



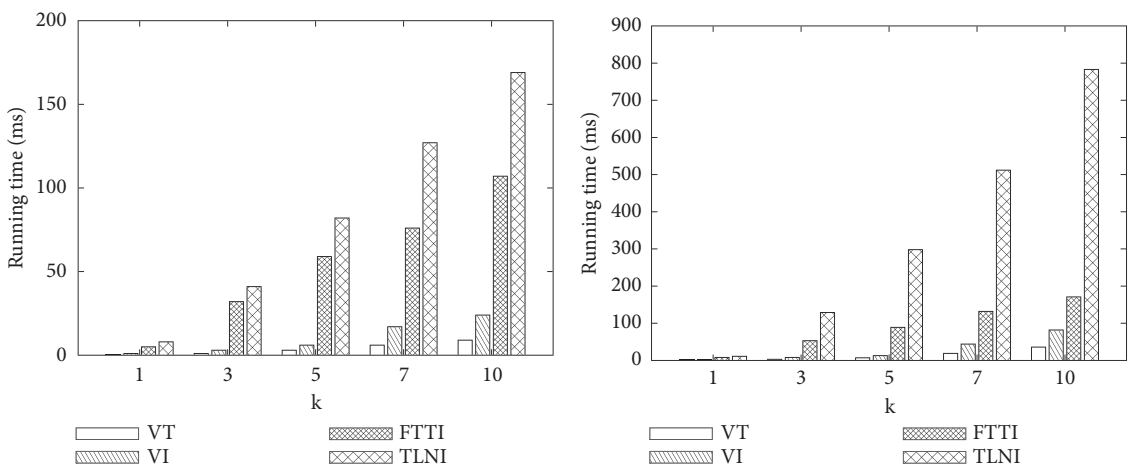
(a) Querying time (b) Index size

Figure 8: Impact of the number of piecewise interval of time function.



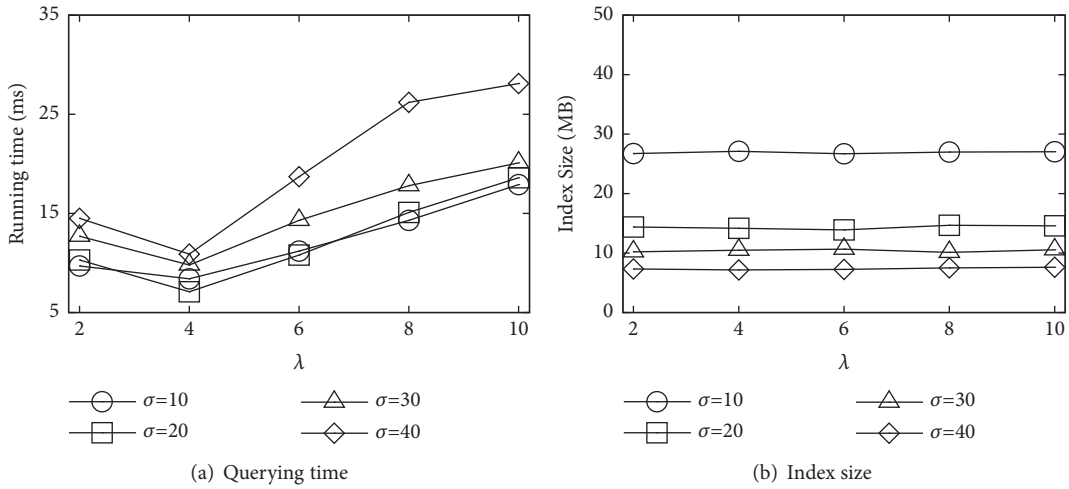
(a) The number of vertices is 100k (b) The number of vertices is 500k

Figure 9: Impact of the average distance to objects.



(a) The number of vertices is 100k (b) The number of vertices is 500k

Figure 10: Impact of k.



(a) Querying time (b) Index size

FIGURE 11: Impact of λ and σ.

these works are based on the estimated value of travel time. However, these indexes cannot facilitate query effectively for the large networks because the deviations are always too large between the estimated and the actual travel time.

Recently, there are some works about the shortest path query between two given vertices over time-dependent graphs [13, 28–33]. However, these works do not study any index that can be used in kNN query over time-dependent road networks. Note that, in [32, 33], the waiting time is incorporated in arrival function $g_i(t)$ which represents the earliest arrival time for every vertex v_i when the departure time is t. Different to these works, we propose nwt-function $w^*_{i,j}(t)$ to represent the minimum traveling time for every edge when the waiting time is allowed. The arrival function $g_i(t)$ cannot be utilized in our voronoi-based index for k nearest objects problem. Otherwise, the intuitive meaning behind $w^*_{i,j}(t)$ is more easily comprehended than $g_i(t)$.

8. Conclusion

In this paper, we study the problem of k nearest object query on time-dependent road networks. We first give an algorithm for processing time-dependent road networks such that the waiting time is not necessary to be considered and then propose a novel voronoi-based index to facilitate kNN query. We explain how to construct the index and complete the querying process using our index. Furthermore, we propose a novel balanced tree, named V-tree, which is a secondary level index on voronoi-based index to make our querying algorithm more efficient. We confirm the efficiency of our method through extensive experiments on real-life datasets.

Data Availability

The road network dataset used to support the findings of this study is included within the article. It can be downloaded from http://snap.stanford.edu/data/roadNet-CA.html.

Additional Points

Different to our previous work published on APWeb-WAIM 2018, in this paper, we propose a novel balanced tree structure, named V-tree, which can be considered as a secondary level index on the voronoi-based index to make k nearest object query more efficient.

Disclosure

An early version of this work has been published in the 2nd Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data (APWeb-WAIM 2018).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China Grant nos. 61402323, 61572353, and U1736103, the Opening Project of State Key Laboratory of Digital Publishing Technology, and the Australian Research Council Discovery Grant DP130103051.

References

[1] T. Abeywickrama, M. A. Cheema, and D. Taniar, "k-nearest neighbors on road networks: A journey in experimentation and in memory implementation," in *Proceedings of the 42nd International Conference on Very Large Data Bases, VLDB 2016*, vol. 9, no. 6, pp. 492–503, September 2016.

[2] K. C. Lee, W. Lee, and B. Zheng, "Fast object search on road networks," in *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, p. 1018, Saint Petersburg, Russia, March 2009.

[3] S. Luo, B. Kao, G. Li, J. Hu, R. Cheng, and Y. Zheng, "TOAIN, a throughput optimizing adaptive index for answering dynamic knn queries on road networks," *The Proceedings of the VLDB Endowment (PVLDB)*, vol. 11, no. 5, pp. 594–606, 2018.

[4] F. Wei-Kleiner, "Finding nearest neighbors in road networks: a tree decomposition method," in *Proceedings of the the Joint EDBT/ICDT 2013 Workshops*, pp. 233–240, Genoa, Italy, March 2013.

[5] M. R. R. B. Chucre, S. M. do Nascimento, J. A. de Macêdo et al., "Taxi, please! a nearest neighbor query in time-dependent road networks," in *Proceedings of the 2016 17th IEEE International Conference on Mobile Data Management (MDM)*, pp. 180–185, June 2016.

[6] B. George and S. Shekhar, "Time-aggregated graphs for modeling spatio-temporal networks," *Journal on Data Semantics*, vol. 11, pp. 191–212, 2006.

[7] E. Kanoulas, Y. Du, T. Xia, and D. Zhang, "Finding fastest paths on a road network with speed patterns," in *Proceedings of the 22nd International Conference on Data Engineering*, p. 10, April 2006.

[8] Y. Li, H. Caö, and Y. Tan, "Novel method of identifying time series based on network graphs," *Complexity*, vol. 17, no. 1, pp. 13–34, 2011.

[9] U. Demiryurek, F. Banaei-Kashani, and C. Shahabi, "Efficient K-nearest neighbor search in time-dependent spatial networks," in *Database and Expert Systems Applications*, vol. 6261 of *Lecture Notes in Computer Science*, pp. 432–449, Springer, Berlin, Germany, 2010.

[10] L. A. Cruz, F. Lettich, L. S. Júnior, R. P. Magalhães, and J. A. F. de Macêdo, "Finding the nearest service provider on time-dependent road networks," in *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery*, pp. 21–31, 2017.

[11] Y. Komai, D. H. Nguyen, T. Hara, and S. Nishio, "KNN search utilizing index of the minimum road travel time in time-dependent road networks," in *Proceedings of the 2014 IEEE 33rd International Symposium on Reliable Distributed Systems Workshops, SRDSW 2014*, pp. 131–137, October 2014.

[12] L. A. Cruz, M. A. Nascimento, and J. A. F. de Macêdo, "k-nearest neighbors queries in time-dependent road networks," *JIDM*, vol. 3, no. 3, pp. 211–226, 2012.

[13] B. Ding, J. X. Yu, and L. Qin, "Finding time-dependent shortest paths over large graphs," in *Proceedings of the 11th International Conference on Extending Database Technology: Advances in Database Technology*, pp. 205–216, Nantes, France, March 2008.

[14] R. Zhong, G. Li, K. Tan, and L. Zhou, "G-tree: an efficient index for KNN search on road networks," in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pp. 39–48, San Francisco, Calif, USA, October 2013.

[15] A. Abou-Rjeili and G. Karypis, "Multilevel algorithms for partitioning power-law graphs," in *Proceedings of the 20th IEEE International Parallel and Distributed Processing Symposium, IPDPS 2006*, April 2006.

[16] I. S. Dhillon, Y. Guan, and B. Kulis, "Weighted graph cuts without eigenvectors a multilevel approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 11, pp. 1944–1957, 2007.

[17] X. Xu, N. Yuruk, Z. Feng, and T. A. Schweiger, "SCAN: a structural clustering algorithm for networks," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery And Data Mining*, pp. 824–833, San Jose, Calif, USA, August 2007.

[18] H. Hu, D. L. Lee, and J. Xu, "Fast nearest neighbor search on road networks," in *Advances in Database Technology - EDBT 2006*, vol. 3896 of *Lecture Notes in Computer Science*, pp. 186–203, Springer, Berlin, Germany, 2006.

[19] X. Huang, C. S. Jensen, and S. Šaltenis, "The islands approach to nearest neighbor querying in spatial networks," in *Advances in Spatial and Temporal Databases*, vol. 3633 of *Lecture Notes in Computer Science*, pp. 73–90, Springer, Berlin, Germany, 2005.

[20] M. R. Kolahdouzan and C. Shahabi, "Voronoi-based K nearest neighbor search for spatial network databases," in *Proceedings of the Thirtieth International Conference on Very Large Data Bases*, pp. 840–851, 2004.

[21] C.-L. Li, E. T. Wang, G.-J. Huang, and A. L. P. Chen, "Top-n query processing in spatial databases considering bi-chromatic reverse k-nearest neighbors," *Information Systems*, vol. 42, pp. 123–138, 2014.

[22] Y. Liu, Z. Li, and X. Zhen, "Empirical study on indicators selection model based on nonparametric k-nearest neighbor identification and R clustering analysis," *Complexity*, vol. 2018, Article ID 2067065, 9 pages, 2018.

[23] S. Yang, M. A. Cheema, X. Lin, Y. Zhang, and W. Zhang, "Reverse k nearest neighbors queries and spatial reverse top-k queries," *The VLDB Journal*, vol. 26, no. 2, pp. 151–176, 2017.

[24] Y. Zheng, Q. Guo, A. K. H. Tung, and S. Wu, "Lazylsh: approximate nearest neighbor search for multiple distance functions with a single index," in *Proceedings of the 2016 ACM SIGMOD International Conference on Management of Data, SIGMOD 2016*, pp. 2023–2037, July 2016.

[25] H. Zhu, X. Yang, B. Wang, and W.-C. Lee, "Range-based obstructed nearest neighbor queries," in *Proceedings of the 2016 ACM SIGMOD International Conference on Management of Data, SIGMOD 2016*, pp. 2053–2068, July 2016.

[26] C. F. Costa, M. A. Nascimento, J. A. Macedo, and J. Machado, "A$*$-based solutions for KNN queries with operating time constraints in time-dependent road networks," in *Proceedings of the 2014 15th IEEE International Conference on Mobile Data Management (MDM)*, pp. 23–32, Brisbane, Australia, July 2014.

[27] C. F. Costa, J. Machado, M. A. Nascimento, and J. A. Macêdo, "Aggregate k-nearest neighbors queries in time-dependent road networks," in *Proceedings of the the 4th ACM SIGSPATIAL International Workshop*, pp. 3–12, November 2015.

[28] J. Kim, W.-S. Han, J. Oh, S. Kim, and H. Yu, "Processing time-dependent shortest path queries without pre-computed speed information on road networks," *Information Sciences*, vol. 255, pp. 135–154, 2014.

[29] J. Ma, B. Yao, X. Gao, Y. Shen, and M. Guo, "Top-k critical vertices query on shortest path," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 10, pp. 1999–2012, 2018.

[30] Y. Yang, H. Gao, J. X. Yu, and J. Li, "Finding the cost-optimal path with time constraint over time-dependent graphs," *Proceedings of the VLDB Endowment*, vol. 7, no. 9, pp. 673–684, 2014.

[31] D. Zhang, D. Yang, Y. Wang, K.-L. Tan, J. Cao, and H. T. Shen, "Distributed shortest path query processing on dynamic road networks," *The VLDB Journal*, vol. 26, no. 3, pp. 399–419, 2017.

[32] L. Li, W. Hua, X. Du, and X. Zhou, "Minimal on-road time route scheduling on time-dependent graphs," *Proceedings of the VLDB Endowment*, vol. 10, no. 11, pp. 1274–1285, 2017.

[33] L. Li, K. Zheng, S. Wang, W. Hua, and X. Zhou, "Go slow to go fast: minimal on-road time route scheduling with parking facilities using historical trajectory," *The VLDB Journal*, vol. 27, no. 3, pp. 321–345, 2018.

Hindawi
Complexity
Volume 2019, Article ID 5370961, 15 pages
https://doi.org/10.1155/2019/5370961

Research Article

Evaluation of Residential Housing Prices on the Internet: Data Pitfalls

Ming Li,[1] **Guojun Zhang**,[2] **Yunliang Chen**,[3] **and Chunshan Zhou**[1]

[1]*School of Geography and Planning, Sun Yat-sen University, Guangzhou 510275, China*
[2]*School of Public Policy and Management, Guangdong University of Finance and Economics, Guangzhou 510275, China*
[3]*School of Computer Science, China University of Geosciences, Wuhan 430074, China*

Correspondence should be addressed to Guojun Zhang; guojunz@gdufe.edu.cn and Yunliang Chen; Cyl_king@hotmail.com

Received 29 November 2018; Accepted 27 January 2019; Published 19 February 2019

Guest Editor: Ke Deng

Many studies have used housing prices on the Internet real estate information platforms as data sources, but platforms differ in the nature and quality of the data they release. However, few studies have analysed these differences or their effect on research. In this study, second-hand neighbourhood housing prices and information on five online real estate information platforms in Guangzhou, China, were comparatively analysed and the performance of neighbourhoods' raw information from four for-profit online real estate information platforms was evaluated by applying the same housing price model. The comparison results show that the official second-hand residential housing prices at city and district level are generally lower than those issued on four for-profit real estate websites. The same second-hand neighbourhood housing prices are similar across each of the four for-profit real estate websites due to cross-referencing among real estate websites. The differences of housing prices in the central city area are significantly fewer than those in the periphery. The variation of each neighbourhood's housing prices on each website decreases gradually from the city centre to the periphery, but the relative variation stays stable. The results of the four hedonic models have some inconsistencies with other studies' findings, demonstrating that errors exist in raw information on neighbourhoods taken from Internet platforms. These results remind researchers to choose housing price data sources cautiously and that raw information on neighbourhoods from Internet platforms should be appropriately cleaned.

1. Introduction

Housing sale price statistics for 70 large and medium-sized cities released in December 2016 by the National Bureau of Statistics of the People's Republic of China revealed that, in December, prices of newly constructed housing in megacities had not changed from the previous month. Prices of newly constructed houses in provincial capitals and other large cities rose by 0.2% compared with the previous month, and prices in medium-sized cities increased by 0.4%. According to public opinion, these price levels are underestimated, and this triggers media and public discussion of the accuracy of housing statistics.

Property agents emerged after housing market reforms were implemented in China [1]. With the development of the Internet economy, real estate agency websites have been established. These websites provide masses of information and neighbourhood descriptive information for the renting and selling of residential property, constituting a location-aware form of big data [2]. Data from real estate agency websites has served as a valuable data source for scholars. Research content and results are diverse. Researchers use online housing price data to investigate the determinants of housing prices, relevant policy, and macroeconomic and social situations, such as tax policy, stamp duty [3], housing purchase restriction policy [4], institutional mediation [5], and disease [6]. Structural attributes, such as gross floor area, storey level [7], age of properties [8], and differentials between large-scale estates and single-block buildings [9] and location attributes, such as metro services [10], green space [11, 12], neighbouring and environmental effects [13], and the effects of theme parks on local areas [14], were all investigated by using online housing price data. Moreover, online housing price data are employed to explain various

phenomena in the housing market, such as the spatiotemporal trends concerning housing price fluctuations [15], the spatial pattern of rent prices [16], the transmission of house price changes across quality tiers [17], the *housing ladder effect* [18], buyers' preferences for high-end residential property [19], and corruption in China's land market auctions [20]. Moreover, housing prices and neighbourhood information on real estate broker websites can be used as input variables for other models. Housing prices have been considered as influential factors when simulating urban growth [21], and neighbourhood data obtained from the Lianjia website have been used to create the Urban Form Index [22]. Housing prices on Internet information platforms have been extensively used in housing market research.

Internet real estate data have several advantages. First, users share housing information in a timely manner according to their own interests and they are willing to update this information. Internet real estate agency platforms can either hire their own agents or rent out some interfaces to other real estate agencies who can share their own property information. Furthermore, leasers can register accounts and list their own properties on these websites. The second advantage of Internet real estate data is that the cost of data acquisition is comparatively low. Most of the cost is paid by traditional real estate agencies. They gather and organize the data. The third advantage is that Internet real estate data are detailed. Users provide the location, type, structure, construction time, renovation pictures, and videos of an apartment or house. Some websites even provide information about local facilities, such as bus stops, supermarkets, hospitals, kindergartens, and subway stations. In addition, real estate agency websites document housing prices on different scales, at the city, district, subdistrict, and neighbourhood levels, as well as for individual houses and apartments.

However, as a type of big data, data on the real estate agency websites share the same defects. Sampling errors, measurement errors, aggregation errors, and errors associated with the systematic exclusion of information also exists [23]. The first reason for this is that the sampling process is biased. Due to the commercialization of real estate agencies, they do not tend to invest resources in areas where the market is small and the profit margins are low, such as suburban areas. Information density in developing areas is low, which may even lead to data blind zones. Furthermore, Internet housing data lack systematic validation. Some real estate agents may falsify lower housing prices to attract renters. Website operators and relevant government departments may have difficulty supervising such behaviour. Another reason for the inaccuracy of Internet housing data is the duplication of housing information. Different real estate agents may issue the same housing on the website. Each website calculates housing prices using their own property databases, and such problems certainly introduce errors to housing prices.

Although the accuracy of residential property prices is an essential foundation of real estate research and significant gaps exist between official housing prices and housing prices issued by each real estate agency website, the differences in various housing price data sources are likely to be overlooked. Much research uses housing prices without checking data reliability. Research about the quality of real estate price data products has mainly focused on various house price indexes [24–27]. Although house price indexes are crucial for academic research to more thoroughly understand the housing market, house price indexes are not intuitive for a public that lacks relevant background knowledge. Moreover, much research uses housing prices rather than price indexes as a data source [28]. How many differences exist among various property price data sources and to what extent these differences affect research are not yet known.

Accurate house prices are of theoretical importance and are crucial to understanding the operation of the housing market. Therefore, the primary objective of this study is to analyse differences among housing prices on mainstream online real estate information platforms and to evaluate the performance of neighbourhoods' raw information from for-profit online real estate information platforms by applying the same housing price model. Housing price data at the city and district levels from five Internet real estate information platforms were collected and compared. Then, second-hand neighbourhoods' housing prices from four for-profit Internet real estate information platforms were compared. Finally, raw information on neighbourhoods from the four platforms, including housing prices and the construction year, was put into the same hedonic housing price model to evaluate the performance of data on each platform (Figure 1). If results from the model contradicted other studies, the raw input housing information data were assumed to be unreliable.

2. China's Internet Real Estate Information Platforms

China's online real estate information platforms can be divided into four categories [29].

(1) Internet platforms for traditional bricks-and-mortar real estate agency firms: these websites are established by traditional real estate agency firms to promote their housing resources online. These websites serve mainly as a property database where agents and renters can search for housing information. Then, renter contact agents directly and continue the transaction offline. Typical platforms include Centaline Property, Lianjia, and Q Fang. Centaline Property (http://www.centanet.com), which has approximately 2,000 branches and over 60,000 employees in China, was selected. Centaline Property enjoys the largest portion of the Guangzhou real estate market. Lianjia.com (http://www.lianjia.com), the online platform of Beijing Lianjia Real Estate Brokerage Co., Ltd., was selected as a data source. Lianjia has approximately 8,000 stores and more than 1.3 million agents. Lianjia acquired My Top Home to enter the market in the Pearl River Delta in 2015.

(2) Internet real estate information platforms: these websites do not hire their own agents nor do they open stores. They serve as housing advertisement platforms. Traditional real estate agency firms can pay for their agents to release housing information on them, and individual users can share housing information for free. Moreover, such websites release real estate news and analysis reports. Anjuke, Sohu Focus, Sina Leju, and 58 Tongcheng are representative of such sites.

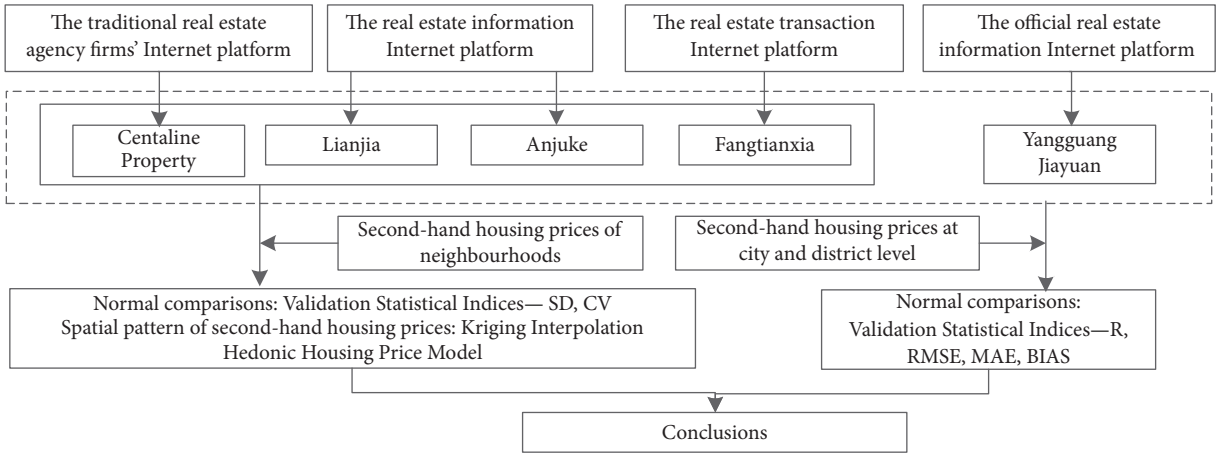


Figure 1: Flowchart of the housing price evaluation process.

Anjuke Inc. (http://www.anjuke.com), whose app has been installed 170 million times, was selected for this study. Its coverage of the market reaches 88%, encompassing 500 cities throughout the country.

(3) Real estate transaction Internet platforms: as for traditional real estate agency firms, such companies open stores and hire agents but not to the extent that Internet-based firms do. In contrast to traditional real estate agency firms' offline-to-online pattern, real estate transaction platforms started as online businesses and expanded offline. They also lend their interfaces to traditional real estate agency firms. Their stores are mainly for providing experiences and advertisements. Fangtianxia (formerly Soufang) is an example of this type of site. Fangtianxia, with more than 42 million registered users in March, 2015, hires approximately 3.7 million agents and covers more than 500 cities in China. It was the leading Internet portal for real estate in China, as measured by the numbers of page views and visitors to its websites in 2014, according to DCCI (http://www.dcci.com.cn).

(4) Official real estate information Internet platforms: relevant government departments, such as the Housing and Urban–Rural Development Bureau, create real estate information websites to release policies, property prices, property resources, and other information. Yangguang Jiayuan (http://www.gzcc.gov.cn/data/) is an official real estate information platform created by the Guangzhou Housing and Urban-Rural Construction Committee. Official statistical data, policies, and housing resources are released on it but its data volume is less than for-profit Internet real estate information platforms'.

3. Study Area and Data

3.1. Study Area. With a resident population of over 14 million in 2016, Guangzhou is the provincial capital of Guangdong Province, the economic and political centre of the Pearl River Delta region, and one of China's megacities. It was selected for the study because it has been at the forefront of reform since the 1980s and was the first provincial capital city to implement comprehensive housing system reform from 1978 [30]. Following many years of administrative division adjustments, Guangzhou now has 11 districts (Figure 2). According to the *Guangzhou Master Plan (2011–2020)*, the central area of Guangzhou contains Yuexiu District, Liwan District, Haizhu District, Tianhe District, the southern part

Figure 2: District Boundary of Guangzhou in 2016.

Table 1: List of the statistical indices for validation used to compare for-profit real estate information website housing prices with the official data.

Validation Statistical Index	Equation	Perfect Value
Pearson correlation coefficient (R)	$R = \frac{\sum_{i=1}^{n}(P_i - \overline{P})(O_i - \overline{O})}{\sqrt{\sum_{i=1}^{n}\left(P_i - \overline{P}\right)^2} \cdot \sqrt{\sum_{i=1}^{n}\left(O_i - \overline{O}\right)^2}}$	1
Root Mean Squared Error (RMSE)	$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(P_i - O_i\right)^2}$	0
Mean Absolute Error (MAE)	$MAE = \frac{1}{n}\sum_{i=1}^{n}\left\|P_i - O_i\right\|$	0
Relative Bias (BIAS)	$BIAS = \frac{\sum_{i=1}^{n}(P_i - O_i)}{\sum_{i=1}^{n} O_i} \times 100\%$	0

Note: number of months is represented by n; P and O are the for-profit real estate information website housing prices and the official data, respectively.

of Baiyun District, the southern part of Huangpu District, and the northern part of Panyu District; other areas belong to the periphery of Guangzhou. The urban spatial structure of Guangzhou has developed to become polycentric. The traditional city centre is Renmin Park, near the municipal government in Yuexiu District, and the new city centre is Zhujiang New Town in Tianhe District [31–33].

3.2. Data. Data used in this study can be divided into real estate data and point of interest (POI) data. The business scope of real estate agency websites involves various types of commercial real estate, including residences, offices, shops, parking lots, and factory buildings. This research focuses on second-hand residential houses, which are closely related to people's livelihoods. Five representative real estate agency websites were selected: Centaline Property, Lianjia, Anjuke, Fangtianxia, and Yangguang Jiayuan. Second-hand residential housing prices at city and district level were collected between May 2015 and May 2016. Yangguang Jiayuan does not release second-hand neighbourhood housing prices. Each neighbourhood's housing prices and construction time on the four for-profit real estate information websites were collected on June 7, 2016, including 9,941 records from Centaline Property, 6,500 records from Lianjia, 8,198 records from Anjuke, and 6,119 records from Fangtianxia.

A point of interest (POI) is a specific point location that may be useful or of interest. It is a type of point datum representing a real geographic entity, including spatial information, such as latitude and longitude, and address; attribute information, such as names and categories, restaurants, stores, cinemas, and theatres. For this study, the locations of subway stations, parks, and key schools in Guangzhou were obtained from a Chinese map website, Gaode Map (http://www.amap.com) in June 2016. The location of each neighbourhood was obtained through the Gaode API (http://lbs.amap.com/console/show/picker).

4. Methodology

4.1. Normal Comparisons. Housing prices released on the official real estate information platform, Yangguang Jiayuan, are used as fiducial market prices in this study. To quantitatively compare for-profit real estate information website housing prices and official data, two types of statistical measure, namely, degree of agreement and error and bias, are used. Degree of agreement is represented by the Pearson correlation coefficient (R), which reflects the degree of linear correlation between the for-profit real estate information website housing prices and official data. In terms of error and bias, three statistical indices for validation were considered: the mean absolute error (MAE) represents the average magnitude of the error. Although the root mean square error ($RMSE$) also measures average error magnitude, it gives greater weight to the larger errors relative to the MAE. Relative bias ($BIAS$) describes the systematic bias of the for-profit real estate information website housing prices. Equations of each index are presented in Table 1.

The statistical indices for validation used to detect the variation in a neighbourhood's housing prices between websites were standard deviation and coefficient of variation. Standard deviation (SD) quantifies the variation or dispersion of a set of data values. The coefficient of variation (CV), also known as relative standard deviation (RSD), demonstrates the extent of relative variability. It expresses the precision and repeatability of a data set. The equations of each index are listed in Table 2.

4.2. Kriging Interpolation. Kriging is one of the optimal linear predictors based on spatial autocorrelation. The Kriging method predicts values on a continuous surface based on observed sampled data [34]. Housing price predictions at unobserved locations require geostatistical approaches, particularly Kriging interpolation. Kriging compares favourably to the ordinary least squares method (OLS) for predicting house prices [35].

4.3. Hedonic Housing Price Model. The hedonic model has been extensively employed in numerous empirical housing market studies and has proven to be effective [36, 37]. This model is therefore used to evaluate the performance of neighbourhoods' raw information from the for-profit online real

Table 2: List of the statistical indices for validation used to detect variations in a neighbourhood's housing prices between websites.

Validation Statistical Index	Equation	Perfect Value
Standard Deviation (SD)	$\mathrm{SD} = \sqrt{\frac{\sum_{i=1}^{m}(p_i - \overline{p})^2}{m}}$	0
Coefficient of Variation (CV)	$\mathrm{CV} = \frac{\sqrt{(\sum_{i=1}^{m}(p_i - \overline{p})^2)/m}}{\overline{p}}$	0

Note: the number of websites is represented by m; p_i is the second-hand neighbourhood housing prices on website i.

Table 3: Descriptions of variables.

Variable		Description
Dependent variable	Price	Log of the second-hand neighbourhood's housing prices (¥/Chinese Yuan)
Independent variables	Year	The year of neighbourhood construction (a)
	Centre	Log of distance from the neighbourhood to the nearest city centre (m)
	Subway	Log of distance from the neighbourhood to the nearest subway station (m)
	School	Log of distance from the neighbourhood to the nearest key school (m)
	Park	Log of distance from the neighbourhood to the nearest park (m)

estate information platforms instead of other less common and more complicated models.

The hedonic model is based on Lancaster's [38] consumption theory. Goods are assumed to possess multiple characteristics in fixed proportions, and these characteristics—not the goods themselves—are assumed to dictate consumers' preferences. Rosen [39] developed market equilibrium theory. The aim of the hedonic pricing model is to assess the relationship between the market value of a composite good and each single attribute by generating a set of implicit prices for all these attributes. In general, housing price can be classified as [40]

$$P = f(S, L, N) \tag{1}$$

where P is the market price of a neighbourhood; S is structural attributes, such as construction time, building materials, and ratio of green space; L is location attributes, such as the distance to the city centre, shopping centre, and the nearest subway stations; and N is neighbourhood attributes, for example, school quality, environment quality, and natural scenery.

The three equation types most often used for hedonic price models are pure linear, semilog, and log–log. The two log forms are more appropriate than the linear form is, because the law of diminishing marginal utility applies to the situation. Coefficients of the log-log form are the percentage change in market price in response to a 1% change in each attribute's implicit prices. The log-log form was employed in this study. It can be defined as

$$\ln P = \alpha + \sum_{i=1}^{n} \beta_i \ln C_i + z \tag{2}$$

where P represents the market price of a neighbourhood; C represents the quantity of utilities or services that the house provides, concerning house characteristics and nearby infrastructure; for example, β is the regression coefficient of the study variables; α is the constant; z is the random error term; and n is the number of neighbourhoods [36].

Second-hand housing price data and neighbourhood information from the four for-profit real estate information websites are entered into the same hedonic house price model to test whether the use of different raw data sources would affect the modelling outcomes.

The descriptions of the explanatory variables that are used in the hedonic models are listed in Table 3. Neighbourhoods' property prices and the years in which they were constructed are obtained from Centaline Property, Lianjia, Anjuke, and Fangtianxia, respectively. *Price* is the log form of a neighbourhood's average second-hand housing price.

Housing prices are observed to have a negative relationship with age [8]. *Year* is the year when a neighbourhood was constructed. Location is widely recognized to be the primary determinant of housing price. The distance to the city centre accounts for a substantial proportion of variations in housing prices, which corresponds with the predictions of the bid-rent curve of renting prices [41]. Because Guangzhou is a polycentric city [31], the People's Government of Guangzhou Municipality and Zhujiang New Town are selected as the two city centres. *Centre* is the log form of distance from the neighbourhood to the nearest city centre. Most studies have concluded that the proximity of housing to subway stations positively affected value [10]. The subway station list used for this study was obtained from Guangzhou Metro (http://www.gzmtr.com/). *Subway* is the log form of the distance from the neighbourhood to the nearest subway station. The nearby enrolment policy and the school district system have been implemented since 1986 in China's compulsory education system. Key public schools have significant effects on housing prices [42]. The key school list used for this study was obtained from the Education Bureau of Guangzhou (http://www.gzedu.gov.cn/), and *School* is the log form of the distance from a neighbourhood to the nearest key school. A park is a major green space with ecological, entertainment,

Table 4: Validation indices for city-level second-hand housing price data.

Real Estate Agency Websites	R	RMSE	MAE	BIAS
Centaline Property	0.737	10222.360	10197.920	72.262
Lianjia	0.836	5772.276	5736.600	40.649
Anjuke	0.943	6192.564	6173.846	43.747
Fangtianxia	0.950	6384.281	6374.231	54.008

Notes: units for RMSE, MAE, and BIAS are CNY.

social, and cultural functionality. Therefore, other studies have concluded that house prices increase with increasing proximity to nearby parks [43]. For this study, parks in Guangzhou were found using Gaode Map. *Park* is the log form of the distance from a neighbourhood to the nearest park. The locations of neighbourhoods, subway stations, and key schools were obtained from the Gaode Map API. Distances were calculated based on their coordinates.

5. Comparison Results and Discussions

5.1. Second-Hand Housing Prices at City Level. The second-hand residential housing price trends from May 2015 to May 2016 are shown in Figure 3. The official second-hand residential housing prices released on Yangguang Jiayuan are significantly lower than those on the four for-profit real estate information websites. All the data present a steady upward trend in second-hand residential housing prices in Guangzhou. The price data released by Centaline Property are substantially higher than the other websites' data and they fluctuate dramatically. They also follow an upward trend in general. Prices from Lianjia, Anjuke, and Fangtianxia are similar. Their volatility is low and their rises are minimal.

The index values of each for-profit real estate information website's housing prices are listed in Table 4. Although housing prices at city level for Anjuke and Fangtianxia are considerably higher than those for Yangguang Jiayuan, their price data share an extremely similar trend, with correlation coefficients of 0.943 and 0.950, respectively. Second-hand residential housing prices in Guangzhou released by Lianjia and Yangguang Jiayuan also have a high correlation (the correlation coefficient reaches 0.836), whereas the correlation between the data of Centaline Property and Yangguang Jiayuan is relatively lower.

Several reasons could be suggested for why the official second-hand residential housing prices released on Yangguang Jiayuan are significantly lower than those on the four for-profit real estate information websites. (1) The final deal price is, in general, higher than the original offer price in second-hand residential housing transactions, because buyers usually bargain with landlords. Housing prices on Yangguang Jiayuan are based on the contract submitted to the relevant administrative housing department, namely, the final deal price; for-profit real estate information website housing prices are based on real estate agents' own databases. The quoted price and the final price are all included in the housing price model. Some transaction records may take the original offer price as the final deal price. (2) Real estate agency firms are for-profit. They do not tend to invest resources into lower priced or marginal regions, such as urban villages or suburban areas, such as Nansha District and Conghua District. Their agency branches are mostly distributed in district centres, where housing prices are higher than in other areas; however, the official data include all transaction records, meaning that many low-priced housing transactions are included. For example, Centaline Property has no branches in Nansha District or Conghua District. This situation magnifies the gap between official and for-profit real estate information websites' second-hand residential housing prices. (3) Tax evasion occurs. Some buyers sign twin contracts with the landlord, one of which is at a lower price and is submitted to the relevant administrative housing department to incur less tax [44].

Some variations also exist between price data of different for-profit real estate information websites because their housing resources are different in each district. For instance, Centaline Property has no housing resources in Nansha District or Conghua District, whereas the Anjuke website has 196 and 121 neighbourhoods in Nansha District and Conghua District, respectively.

5.2. Second-Hand Housing Prices at District Level. Guangzhou has undergone many administrative division adjustments. Therefore, variations exist in statistical division for different real estate agency websites. For this study, eight common districts were selected, namely Yuexiu District, Liwan District, Haizhu District, Tianhe District, Baiyun District, Panyu District, Huadu District, and Zengcheng District.

Figure 4 presents five websites' second-hand residential housing prices for eight districts in Guangzhou between May 2015 and May 2016. The official second-hand residential housing prices of each district are evidently still significantly lower than those from the for-profit real estate information websites, and the volatility of the official price data at district level is higher than that at city level. The overall trend is rising. The second-hand residential housing prices of Centaline Property in the eight respective districts are relatively high. Because housing resources in peripheral areas such as Huadu District and Zengcheng District are fewer than those in central areas, the second-hand residential housing prices of Centaline Property in these two districts are volatile. The second-hand residential housing prices of Lianjia in each district are the least volatile and maintain a gentle upward trend. Except in Yuexiu District, the gaps in second-hand residential housing prices among five websites are significant, both in suburban and urban districts. Although the housing prices of Lianjia, Anjuke, and Fangtianxia are at the same

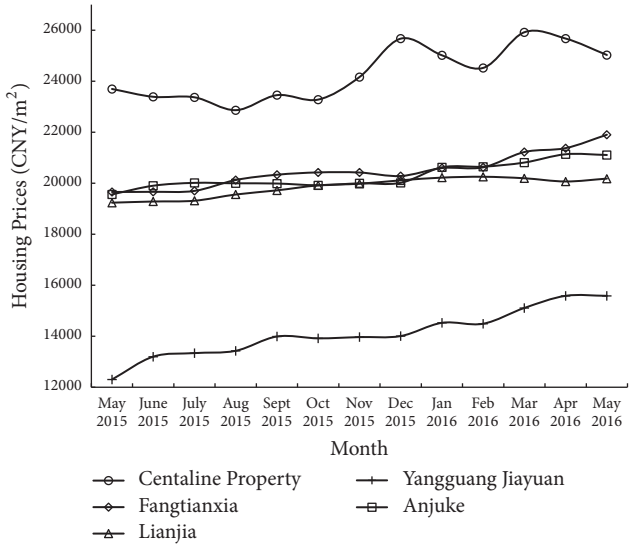


Figure 3: Second-hand residential housing prices in Guangzhou (May 2015-May 2016).

moderate level in Huadu District, Baiyun District, and Panyu District, those of Centaline Property are always higher and those of Yangguang Jiayuan are significant lower.

In general, the volatility of district-level prices is higher than that of city-level prices because of the smaller sample size at district level. In terms of prices, the official data of each district are still significantly lower, and Centaline Property's prices are at a comparatively high level for most districts. The gaps of second-hand residential housing prices among the five websites between prices do not vary significantly from suburban to urban districts.

5.3. Second-Hand Housing Prices of Neighbourhoods. Because Yangguang Jiayuan does not release the housing prices of each neighbourhood, only second-hand neighbourhoods' housing prices from four for-profit real estate information websites'—namely Centaline Property, Lianjia, Anjuke, and Fangtianxia—were collected. Then, 1,897 common neighbourhoods were selected. If each neighbourhood's housing prices were equal on the four websites, scatter plot points would be distributed on the $y = x$ line. Each neighbourhood's second-hand housing prices were compared pairwise on each of the four agency websites, and the results are given in Figure 5.

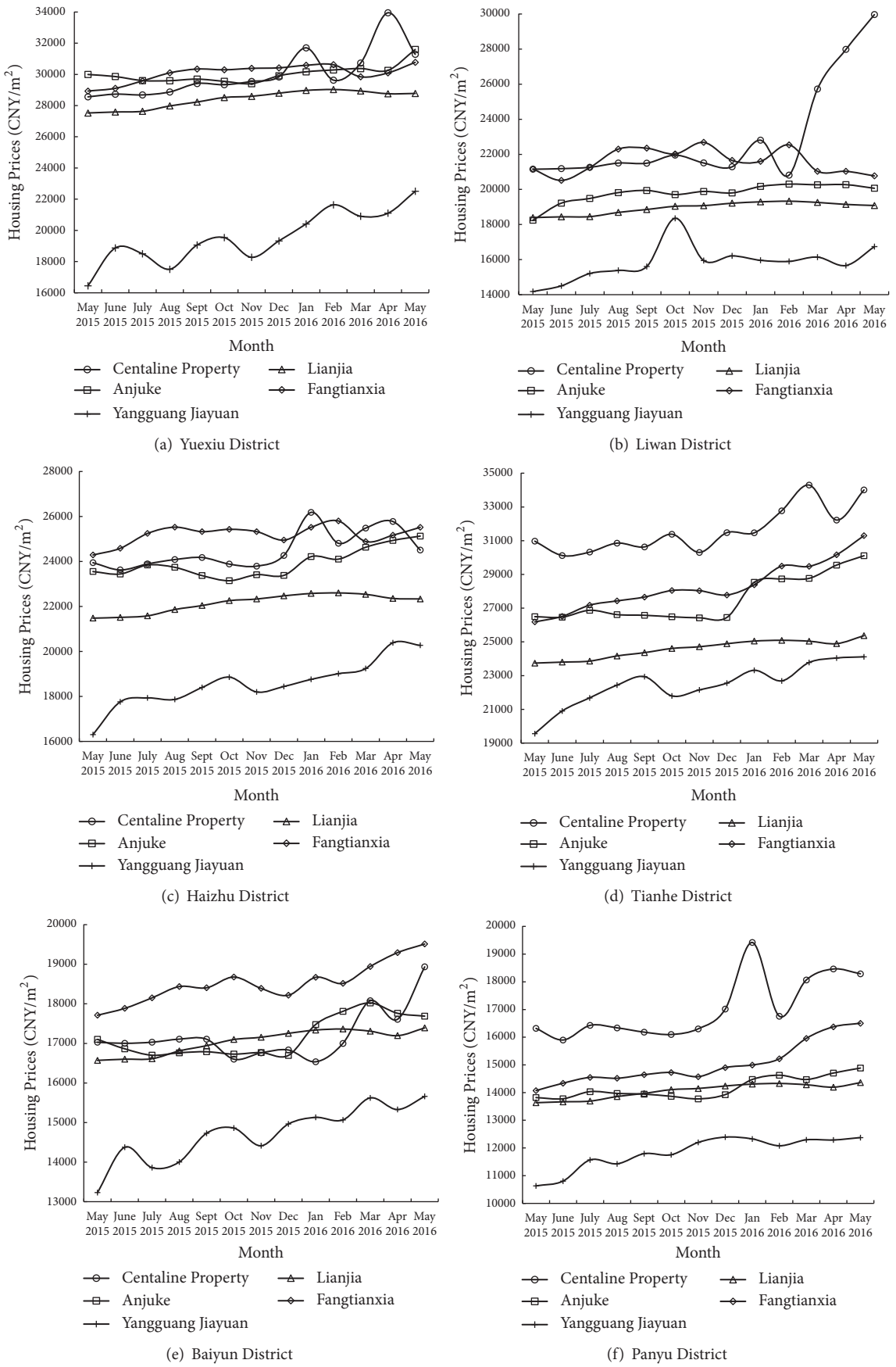
A pairwise comparison illustrates, as in Figure 5, that each neighbourhood's housing prices are similar on different for-profit real estate information websites. Moreover, each neighbourhood's housing prices demonstrate consistency between Centaline Property and Fangtianxia data. Consultations with real estate agency staff revealed that, in the real estate agency industry, second-hand neighbourhoods' housing prices are not only calculated with their own databases but can also be artificially modified using other real estate agency websites' price data. Therefore, one neighbourhood's housing prices can be approximately similar on different websites.

The same neighbourhood's housing prices on the four for-profit real estate information websites can be treated as one group. The *SD* and *CV* of each group were calculated. *SD* was used to quantify the variation of second-hand neighbourhoods' housing prices on different websites, and *CV* was used to express the relative variation. Kriging interpolation was used to detect the spatial distribution features of *SD* and *CV* for second-hand neighbourhoods' housing prices on different websites, and the interpolation results are provided in Figure 6.

The variation of each neighbourhood's housing prices on each website decreases gradually from the city centre to the periphery (Figure 6(a)), because the second-hand residential housing prices in the central area are higher than those in the suburban areas. However, the relative variation of prices is stable across Guangzhou (Figure 6(b)). The two peaks in the north of Tianhe District and in the south-east of Panyu District are caused by errors in websites' prices after verification, which also proves that errors occur in housing prices on the Internet.

5.4. Spatial Pattern of Second-Hand Residential Housing Prices. For this study, neighbourhoods with housing prices were selected and located using the Gaode Map API. Finally, 9,941 records from Centaline Property, 6,500 records from Lianjia, 8,198 records from Anjuke, and 6,119 records from Fangtianxia were entered into the Kriging interpolation, which was used to analyse the spatial pattern of second-hand residential housing prices in Guangzhou (Figure 7).

Spatial patterns of second-hand residential housing prices obtained using the four websites' data are seen to be similar in the central areas, namely Yuexiu District, Liwan District, Haizhu District, Tianhe District, southern Huangpu District, southern Baiyun District, and northern Panyu District. Second-hand neighbourhoods' housing prices in Zhujiang New Town, Ersha Island, Huijing New Town, Pazhou, and Baiyun Fortress Villa have the highest prices. The differences between the four spatial patterns of second-hand residential housing prices in the peripheral areas, such as Conghua



(a) Yuexiu District

(b) Liwan District

(c) Haizhu District

(d) Tianhe District

(e) Baiyun District

(f) Panyu District

Figure 4: Continued.

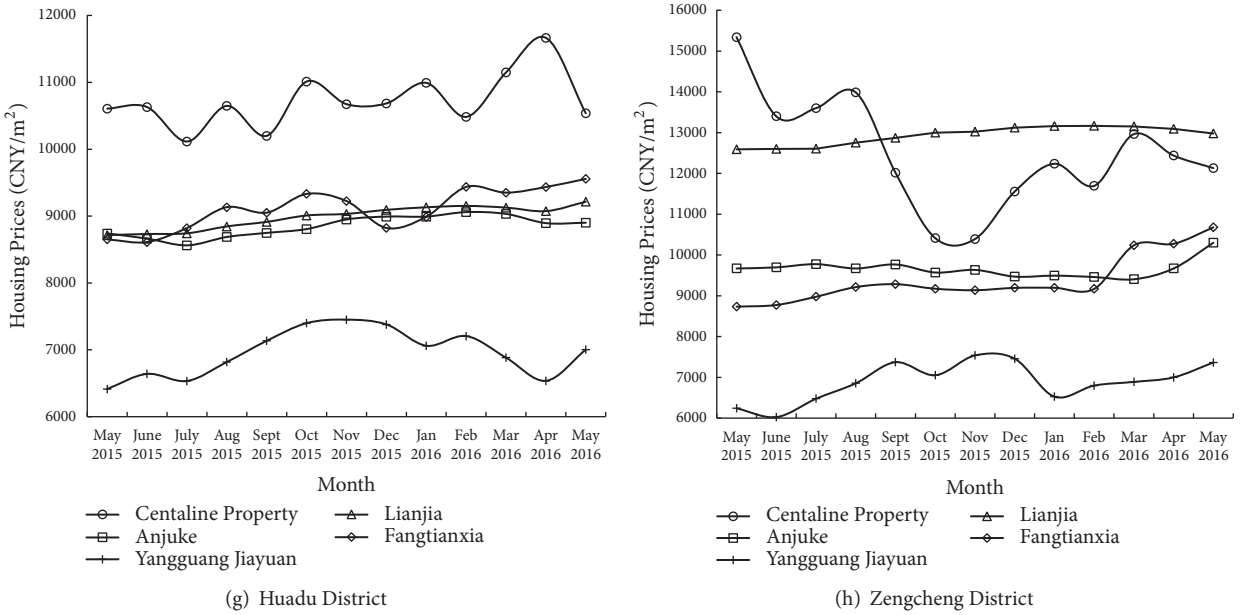


Figure 4: Second-hand residential housing prices of each district (May 2015-May 2016).

District, Zengcheng District, Huadu District, Nansha District, northern Panyu District, northern Baiyun District, and northern Huangpu District are significant. This is because fewer agency branches exist in suburban areas. Real estate agency firms tend to invest more resources in city centres, which leads to fewer peripheral second-hand neighbourhoods being included in the databases. The differences in second-hand neighbourhoods' housing prices on the four for-profit real estate information websites are therefore amplified after spatial interpolation.

5.5. Results of Hedonic Housing Price Models. Because we focused on evaluating the performance of raw data on neighbourhoods from Internet real estate information platforms in housing market research, a classic, reliable, and widely used model—the hedonic housing price model—was selected. If results of the model contradicted those of other studies, the raw input housing information data were assumed to be unreliable.

To maintain data consistency for the four for-profit real estate information websites in question, complex data cleaning was not applied. Neighbourhoods with relatively complete information were selected and used in the hedonic house price model. Statistical variations in second-hand neighbourhoods' housing prices on the four websites are provided in Figure 8. Anjuke has the largest dispersion of second-hand neighbourhoods' housing prices; Lianjia has the smallest. Centaline Property has the largest proportion of second-hand neighbourhood housing prices that qualify as being at the lower level.

The results of the hedonic housing price models using Centaline Property, Lianjia, Anjuke, and Fangtianxia neighbourhood data are listed in Table 5. All the four models' P values are less than 0.001 and F-statistic values are larger than 800. Therefore, all four models are effective. The Fangtianxia model's adjusted R^2 is the highest of the models' values, at 0.530, and the Centaline Property model's adjusted R^2 is the lowest of the models' values at 0.369.

The distance to the nearest city centre [45] and subway station [46] exhibits a significantly negative relationship to second-hand neighbourhoods' housing prices, as in other studies. According to the standardized coefficients of the distance from the neighbourhood to the nearest city centre (-0.516 in Centaline Property, -0.611 in Lianjia, -0.588 in Anjuke and -0.678 in Fangtianxia) and subway station (-0.150 in Centaline Property, -0.102 in Lianjia, -0.138 in Anjuke and -0.119 in Fangtianxia), the distance to the nearest city centre has a more substantial effect on second-hand neighbourhoods' housing prices than the distance to the nearest subway station does. This matches the findings from Shanghai [47] and Hangzhou [48], China. For all models except that of Centaline Property, the year of construction has a significant positive effect on second-hand neighbourhoods' housing prices, which agrees with the results of other studies [49]. The standardized coefficient of the construction year in Anjuke's model (0.119) is much lower than that in Lianjia (0.253) and Fangtianxia (0.231) models, which also reflects the effects, on one model, of using the data of different agencies. The distance to the nearest park has a significant positive effect on second-hand neighbourhoods' housing prices in all four models. This is inconsistent with findings from other research [43]. The distance to the nearest key school is established to have a significant positive effect in the Centaline Property and Lianjia models, which also contradicts other studies [50]. In Fangtianxia model, the distance to the nearest key school had a negative relationship with housing price, but this was not significant.

By building four hedonic housing price models and comparing their results with findings from other research, errors

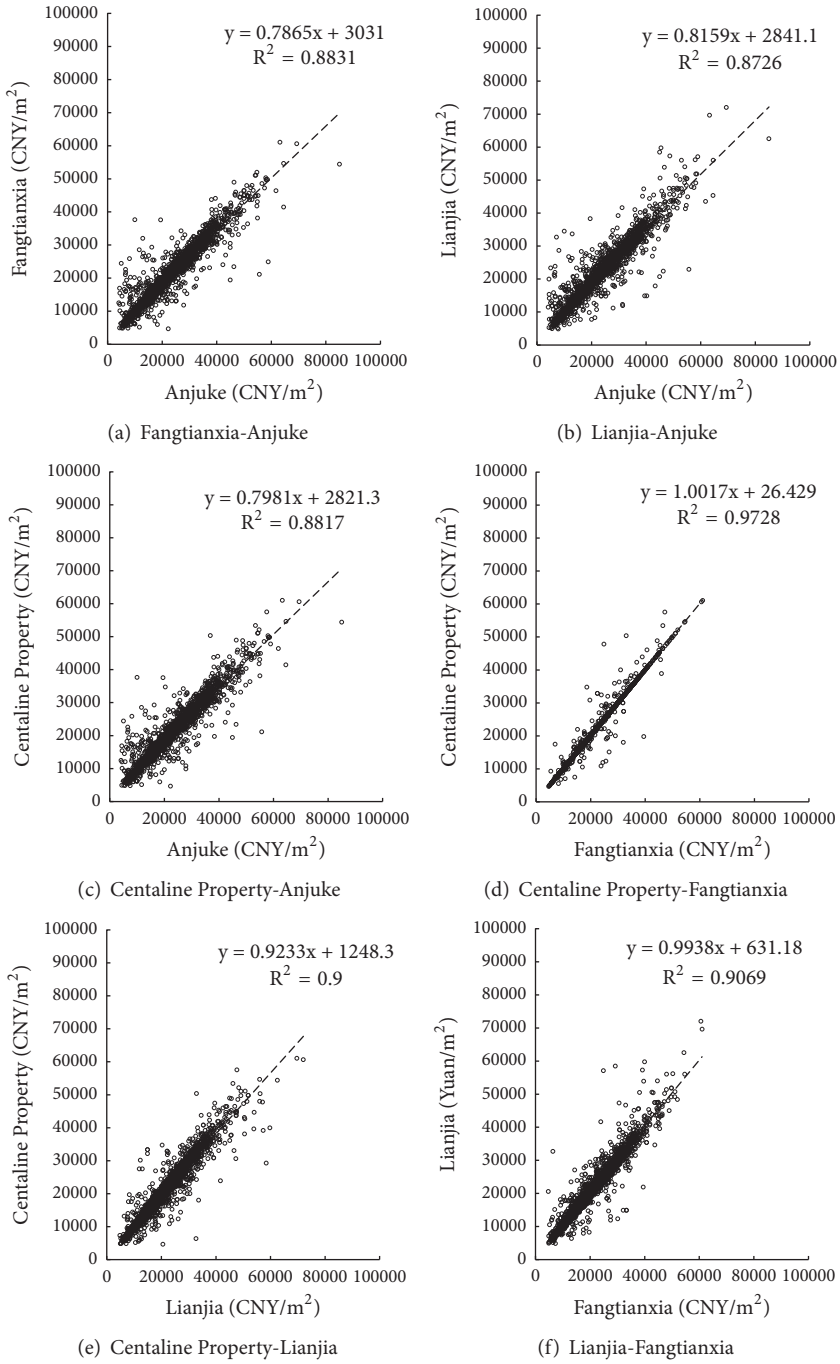
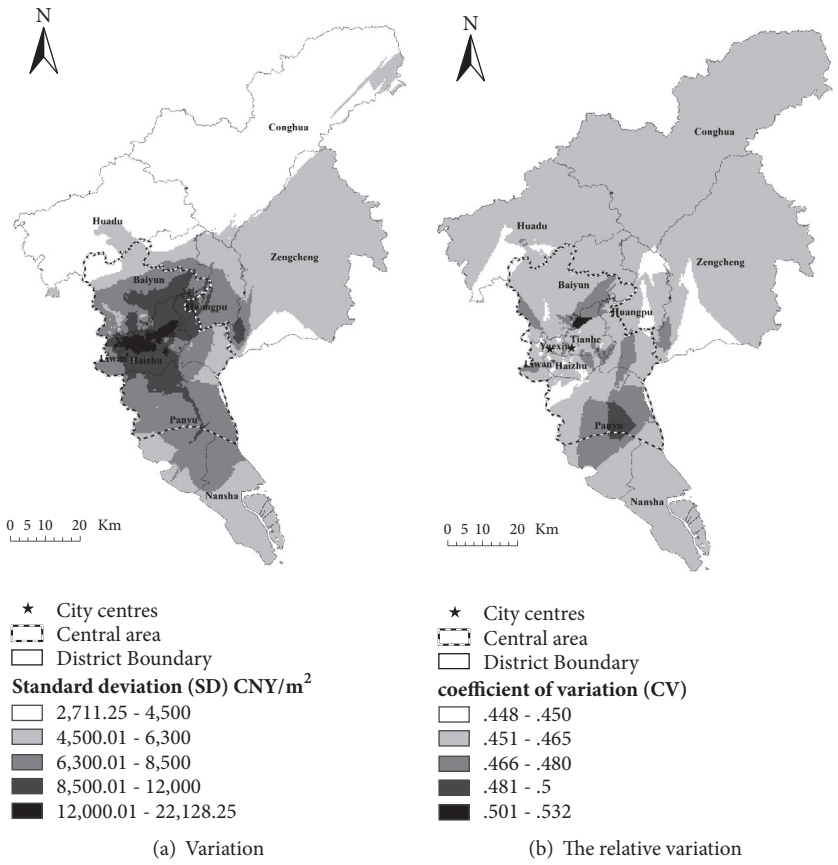


Figure 5: Scatter plots of neighbourhoods' second-hand housing prices (June 7, 2016).

are highlighted in the raw information on neighbourhoods from real estate agency websites, which somewhat affects the accuracy of the housing price models. Some flaws are revealed in the data on construction years for Centaline Property, because property age is proven to negatively correlate with property price [49] which contradicts results from the Centaline Property model. The distance to key school and park are supposed to demonstrate a negative correlation with housing price [43, 50], but this was not the case for results from the four models in the study. Moreover, the differences in the standardized coefficient for the same variables reflect the effects of using different firms' data in a single model. The performance of Fangtianxia model is better than that of other models, because its results match studies more closely. Thus, raw information on neighbourhoods from Fangtianxia are more reliable. But a process of appropriate data cleaning is still essential before we use raw information on neighbourhoods from real estate agency websites.



(a) Variation (b) The relative variation

Figure 6: Spatial interpolation of variation for corresponding second-hand neighbourhoods' housing prices on each website.

Table 5: Results and evaluation of hedonic housing price models.

		Year	Center	Subway	School	Park	Constant
Centaline Property	Coefficient	0.001	-0.253	-0.061	0.015	0.044	10.592
	Standardized Coefficient	0.010	-0.516	-0.150	0.034	0.064	
	Probability	0.236	<0.001	<0.001	0.002	<0.001	<0.001
	R^2: 0.369; adjusted R^2: 0.369; F-statistic: 1163.589; P-value<0.001						
Lianjia	Coefficient	0.016	-0.267	-0.045	0.009	0.043	-20.785
	Standardized Coefficient	0.253	-0.611	-0.102	0.023	0.068	
	Probability	<0.001	<0.001	<0.001	0.061	<0.001	<0.001
	R^2: 0. 451; adjusted R^2: 0. 451; F-statistic: 880.570; P-value<0.001						
Anjuke	Coefficient	0.007	-0.298	-0.059	0.010	0.050	-1.742
	Standardized Coefficient	0.119	-0.588	-0.138	0.022	0.068	
	Probability	<0.001	<0.001	<0.001	0.114	<0.001	0.165
	R^2: 0.449; adjusted R^2: 0. 449; F-statistic: 861.807; P-value<0.001						
Fangtianxia	Coefficient	0.020	-0.321	-0.045	-0.008	0.025	-12.403
	Standardized Coefficient	0.231	-0.678	-0.119	-0.019	0.039	
	Probability	<0.001	<0.001	<0.001	0.259	0.002	<0.001
	R^2: 0.530; adjusted R^2: 0.529; F-statistic: 832.682; P-value<0.001						

6. Conclusions

Housing prices on the Internet are not only a valuable data source for studies but are also commonly used by the public to track real estate market trends. Differences exist among housing prices released on various real estate agency websites, but few studies have compared such data or investigated how much effect differences will exert on relative housing price models. By comparing housing prices in Guangzhou released on official real estate information

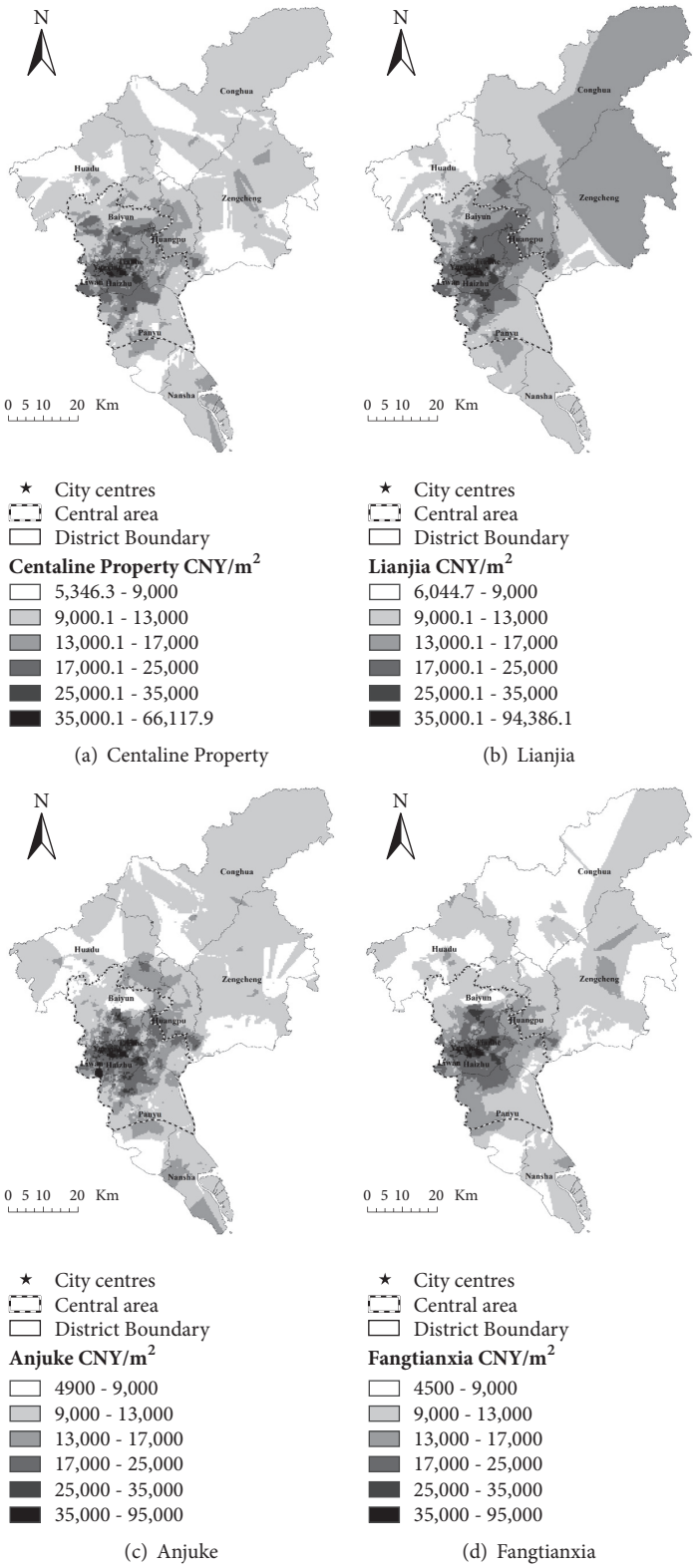


Figure 7: Spatial interpolation of second-hand residential housing prices in Guangzhou.

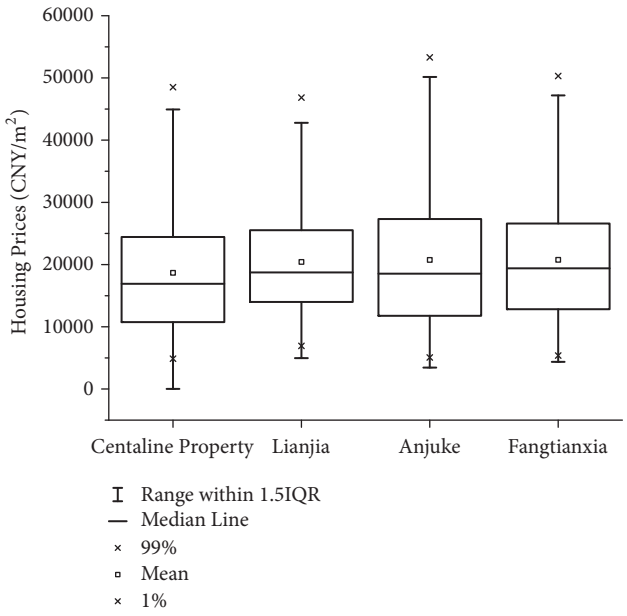


Figure 8: Statistical variations of second-hand neighbourhoods' housing prices on for-profit real estate information platforms.

platform Yangguang Jiayuan and on four for-profit real estate information platforms—namely, Centaline Property, Lianjia, Anjuke, and Fangtianxia—the key results from the analysis are as follows.

(1) The official second-hand residential housing prices in Guangzhou at city level and district level are generally lower than the housing prices issued on for-profit real estate information websites, whereas the overall trend for all types of housing price data is a rising one. Moreover, differences exist among the housing price data of various for-profit real estate information websites. In general, the volatility of district-level prices is higher than that of city-level prices. Data sources should be carefully selected in the study of city-level and district-level housing prices.

The city-level second-hand residential housing prices of Anjuke, Fangtianxia, and Lianjia display a high correlation with the official data. However, Centaline Property's second-hand residential housing prices at city and district level are relatively higher and more volatile than those in official data.

(2) Property prices for corresponding neighbourhoods are similar across the four for-profit real estate information websites as confirmed by cross-referencing. Spatial patterns of second-hand residential housing prices using Kriging interpolation of the four websites' data in Guangzhou are similar in the city centre area, but the differences in the peripheral areas are significant. Housing price variation decreases gradually from the city centre to the periphery, but the relative variation is stable.

(3) The results of the four hedonic models using neighbourhoods' raw information on Centaline Property, Lianjia, Anjuke, and Fangtianxia somewhat contradict other findings. In this study, the distance to the nearest city centre and subway station exhibited a negative relationship with second-hand neighbourhood housing prices, whereas the construction year exhibited a significantly positive relationship, consistent with other studies' findings. However, the distance to the nearest key school and park exerted a positive influence in most models, which was inconsistent with other research findings. Moreover, the differences in standardized coefficient for the same variable demonstrate the effects of different data resources. Fangtianxia model outperformed others. These contradictions and differences demonstrate that errors exist in raw information on neighbourhoods from the Internet, producing incorrect results.

Research shows that differences exist among second-hand residential housing prices of different Internet real estate information platforms at city, district, and neighbourhood levels. Raw information on neighbourhoods from the Internet may be erroneous. Thus, researchers should choose housing price data sources cautiously. Only with appropriate data cleaning can Internet-based information about neighbourhoods be used effectively in studies.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This study is funded by the National Science Foundation of China [41601161].

References

[1] B.-S. Tang, S.-W. Wong, and S.-C. Lui, "Property agents, housing markets and housing services in transitional urban China," *Housing Studies*, vol. 21, no. 6, pp. 799–823, 2006.

[2] Y. Bar-Yam, "From big data to important information," *Complexity*, vol. 21, no. S2, pp. 73–98, 2016.

[3] E. C. M. Hui and C. Liang, "The spatial clustering investment behavior in housing markets," *Land Use Policy*, vol. 42, pp. 7–16, 2015.

[4] Y. Wu and Y. Li, "Impact of government intervention in the housing market: evidence from the housing purchase restriction policy in China," *Applied Economics*, vol. 50, no. 6, pp. 691–705, 2018.

[5] K. K. Fung and R. Forrest, "Institutional mediation, the Hong Kong residential housing market and the Asian Financial Crisis," *Housing Studies*, vol. 17, no. 2, pp. 189–207, 2002.

[6] G. Wong, "Has SARS infected the property market? Evidence from Hong Kong," *Journal of Urban Economics*, vol. 63, no. 1, pp. 74–95, 2008.

[7] E. C. M. Hui, J. W. Zhong, and K. H. Yu, "The impact of landscape views and storey levels on property prices," *Landscape and Urban Planning*, vol. 105, no. 1-2, pp. 86–93, 2012.

[8] H.-G. Kim, K.-C. Hung, and S. Y. Park, "Determinants of housing prices in Hong Kong: a box-cox quantile regression

approach," *The Journal of Real Estate Finance and Economics*, vol. 50, no. 2, pp. 270–287, 2015.

[9] M. H. C. Ho, "Liquidity and price differentials: evidence in the Hong Kong residential re-sale market," *Housing Studies*, vol. 18, no. 5, pp. 745–763, 2003.

[10] S. Li, L. Chen, and P. Zhao, "The impact of metro services on housing prices: a case study from Beijing," *Transportation*, vol. 5, pp. 1–27, 2017.

[11] Y. Xiao, Z. Li, and C. Webster, "Estimating the mediating effect of privately-supplied green space on the relationship between urban public green space and property value: Evidence from Shanghai, China," *Land Use Policy*, vol. 54, pp. 439–447, 2016.

[12] Y. Xiao, Y. Lu, Y. Guo, and Y. Yuan, "Estimating the willingness to pay for green space services in Shanghai: Implications for social equity in urban China," *Urban Forestry & Urban Greening*, vol. 26, pp. 95–103, 2017.

[13] E. C. M. Hui, C. K. Chau, L. Pun, and M. Y. Law, "Measuring the neighboring and environmental effects on residential property value: Using spatial weighting matrix," *Building and Environment*, vol. 42, no. 6, pp. 2333–2343, 2007.

[14] H. Tsai, W.-J. Huang, and Y. Li, "The impact of tourism resources on tourism real estate value," *Asia Pacific Journal of Tourism Research*, vol. 21, no. 10, pp. 1114–1125, 2016.

[15] S. Li, X. Ye, J. Lee, J. Gong, and C. Qin, "Spatiotemporal analysis of housing prices in china: a big data perspective," *Applied Spatial Analysis and Policy*, vol. 10, no. 3, pp. 421–433, 2017.

[16] Y. Chen, X. Liu, X. Li, Y. Liu, and X. Xu, "Mapping the fine-scale spatial pattern of housing rent in the metropolitan area by using online rental listings and ensemble learning," *Applied Geography*, vol. 75, pp. 200–212, 2016.

[17] L. S. Ho, Y. Ma, and D. R. Haurin, "Domino effects within a housing market: The transmission of house price changes across quality tiers," *The Journal of Real Estate Finance and Economics*, vol. 37, no. 4, pp. 299–316, 2008.

[18] L. S. Ho and G. W.-C. Wong, "The first step on the housing ladder: A natural experiment in Hong Kong," *Journal of Housing Economics*, vol. 18, no. 1, pp. 59–67, 2009.

[19] W. M. Jayantha and J. M. Lau, "Buyers' property asset purchase decisions: an empirical study on the high-end residential property market in Hong Kong," *International Journal of Strategic Property Management*, vol. 20, no. 1, pp. 1–16, 2016.

[20] H. Cai, J. V. Henderson, and Q. Zhang, "China's land market auctions: Evidence of corruption?" *The RAND Journal of Economics*, vol. 44, no. 3, pp. 488–521, 2013.

[21] R. Tan, Y. Liu, K. Zhou, L. Jiao, and W. Tang, "A game-theory based agent-cellular model for use in urban growth simulation: A case study of the rapidly urbanizing Wuhan area of central China," *Computers, Environment and Urban Systems*, vol. 49, pp. 15–29, 2015.

[22] J. Wu, N. Ta, Y. Song, J. Lin, and Y. Chai, "Urban form breeds neighborhood vibrancy: a case study using a GPS-based activity survey in suburban Beijing," *Cities*, vol. 74, pp. 100–108, 2018.

[23] R. M. Kaplan, D. A. Chambers, and R. E. Glasgow, "Big data and large sample size: a cautionary note on the potential for bias," *Clinical and Translational Science*, vol. 7, no. 4, pp. 342–346, 2014.

[24] J. H. Mark and M. A. Goldberg, "Alternative housing price indices: an evaluation," *Real Estate Economics*, vol. 12, no. 1, pp. 30–49, 1984.

[25] H. O. Pollakowski, "Data sources for measuring house price," *Journal of Housing Research*, vol. 6, no. 3, pp. 377–387, 1995.

[26] S. C. Bourassa, M. Hoesli, and J. Sun, "A simple alternative house price index method," *Journal of Housing Economics*, vol. 15, no. 1, pp. 80–97, 2006.

[27] Y. M. Goh, G. Costello, and G. Schwann, "Accuracy and robustness of house price index methods," *Housing Studies*, vol. 27, no. 5, pp. 643–666, 2012.

[28] G. Meen, "Regional house prices and the ripple effect: A new interpretation," *Housing Studies*, vol. 14, no. 6, pp. 733–753, 1999.

[29] H. Zhang, "Discussion on present situation and developing tendency of real estate websites in China," *Sci-Tech Information Development and Economy*, vol. 6, pp. 69-70, 2001 (Chinese).

[30] S.-M. Li and Z. Yi, "Financing home purchase in China, with special reference to Guangzhou," *Housing Studies*, vol. 22, no. 3, pp. 409–425, 2007.

[31] F. Wu, "Polycentric urban development and land-use change in a transitional economy: the case of Guangzhou," *Environment and Planning A*, vol. 30, no. 6, pp. 1077–1100, 1998.

[32] L. Jiang and F. Wu, "A study on Guangzhou's employment subcentres and polycentricity," *Urban Planning Forum*, 2009 (Chinese).

[33] J. Liang, "The analysis of the user of the real estate website and suggestion," *China High-Tech Enterprises*, vol. 72, pp. 172–174, 2012 (Chinese).

[34] T. K. Koramaz and V. Dokmeci, "Spatial determinants of housing price values in istanbul," *European Planning Studies*, vol. 20, no. 7, pp. 1221–1237, 2012.

[35] R. A. Dubin, "Predicting house prices using multiple listings data," *The Journal of Real Estate Finance and Economics*, vol. 17, no. 1, pp. 35–59, 1998.

[36] L. Zhang, T. Leonard, and J. C. Murdoch, "Time and distance heterogeneity in the neighborhood spillover effects of foreclosed properties," *Housing Studies*, vol. 31, no. 2, pp. 133–148, 2016.

[37] P. Nilsson, "Are valuations of place-based amenities driven by scale?" *Housing Studies*, vol. 32, no. 4, pp. 449–469, 2017.

[38] K. J. Lancaster, "A new approach to consumer theory," *Journal of Political Economy*, vol. 74, no. 2, pp. 132–157, 1966.

[39] S. Rosen, "Hedonic prices and implicit markets: product differentiation in pure competition," *Journal of Political Economy*, vol. 82, no. 1, pp. 34–55, 1974.

[40] H. M. K. Mok, P. P. K. Chan, and Y.-S. Cho, "A hedonic price model for private properties in Hong Kong," *The Journal of Real Estate Finance and Economics*, vol. 10, no. 1, pp. 37–48, 1995.

[41] J. Chen and Q. Hao, "The impacts of distance to cbd on housing prices in shanghai: a hedonic analysis," *Journal of Chinese Economic and Business Studies*, vol. 6, no. 3, pp. 291–302, 2008.

[42] H. Wen, Y. Xiao, and L. Zhang, "School district, education quality, and housing price: Evidence from a natural experiment in Hangzhou, China," *Cities*, vol. 66, pp. 72–80, 2017.

[43] C. Wu, X. Ye, Q. Du, and P. Luo, "Spatial effects of accessibility to parks on housing prices in Shenzhen, China," *Habitat International*, vol. 63, pp. 45–54, 2017.

[44] J. A. Cao and R. Keivani, "The limits and potentials of the housing market enabling paradigm: an evaluation of China's housing policies from 1998 to 2011," *Housing Studies*, vol. 29, no. 1, pp. 44–68, 2014.

[45] J. Wu, Y. Deng, and H. Liu, "House price index construction in the nascent housing market: the case of china," *The Journal of Real Estate Finance and Economics*, vol. 48, no. 3, pp. 522–545, 2014.

[46] W. Sun, S. Zheng, D. M. Geltner, and R. Wang, "The housing market effects of local home purchase restrictions: evidence from Beijing," *The Journal of Real Estate Finance and Economics*, vol. 55, no. 3, pp. 288–312, 2017.

[47] L. Tan, W. Li, and W. Yu, "A GWR- based study on spatial pattern and structural determinants of Shanghai's housing price," *Economic Geography*, vol. 32, no. 2, pp. 52–58, 2012 (Chinese).

[48] H. Wen and Y. Tao, "Polycentric urban structure and housing price in the transitional China: Evidence from Hangzhou," *Habitat International*, vol. 46, pp. 138–146, 2015.

[49] J. Zietz, E. N. Zietz, and G. S. Sirmans, "Determinants of house prices: a quantile regression approach," *The Journal of Real Estate Finance and Economics*, vol. 37, no. 4, pp. 317–333, 2008.

[50] V. Sah, S. J. Conroy, and A. Narwold, "Estimating school proximity effects on housing prices: the importance of robust spatial controls in hedonic estimations," *The Journal of Real Estate Finance and Economics*, vol. 53, no. 1, pp. 50–76, 2016.

Hindawi
Complexity
Volume 2019, Article ID 8728245, 13 pages
https://doi.org/10.1155/2019/8728245

Research Article

Finding the Shortest Path with Vertex Constraint over Large Graphs

Yajun Yang,[1,2] **Zhongfei Li**,[1] **Xin Wang**,[1] **and Qinghua Hu**[1]

[1]*College of Intelligence and Computing, Tianjin University, China*
[2]*State Key Laboratory of Digital Publishing Technology, Beijing, China*

Correspondence should be addressed to Xin Wang; wangx@tju.edu.cn

Received 30 November 2018; Accepted 31 January 2019; Published 19 February 2019

Guest Editor: Xin Huang

Graph is an important complex network model to describe the relationship among various entities in real applications, including knowledge graph, social network, and traffic network. Shortest path query is an important problem over graphs and has been well studied. This paper studies a special case of the shortest path problem to find the shortest path passing through a set of vertices specified by user, which is NP-hard. Most existing methods calculate all permutations for given vertices and then find the shortest one from these permutations. However, the computational cost is extremely expensive when the size of graph or given set of vertices is large. In this paper, we first propose a novel exact heuristic algorithm in best-first search way and then give two optimizing techniques to improve efficiency. Moreover, we propose an approximate heuristic algorithm in polynomial time for this problem over large graphs. We prove the ratio bound is 3 for our approximate algorithm. We confirm the efficiency of our algorithms by extensive experiments on real-life datasets. The experimental results validate that our algorithms always outperform the existing methods even though the size of graph or given set of vertices is large.

1. Introduction

Graph is an important *complex network model* to describe the relationship among various entities in real applications, including knowledge graph, RDF graph, linked data, social network, biological network, and traffic network [1–4]. Shortest path query is a basic problem on graph model. For example, in knowledge graphs, it is to find the closest connection between two entities or concepts; in social networks, it is to find the closest relationships such as friendship between two individuals; in traffic networks, it is to compute the shortest route between two locations.

Shortest path routing is an important problem in *location-based services* (LBS) and has been well studied in the past decades [5–7]. However, a special kind of shortest path query with vertex constraint is more and more important in real life. For instance, in knowledge graphs, a data miner is interested in investigating the closest relationship between two entities connected by some specified entities or concepts. In traffic networks, carpooling becomes a common business with the rapid development of sharing economy. A car driver may carry some fellows on the way home from company and the fellows are going to get down at distinct locations. Thus a critical problem is how to find a route with the minimum length passing through these locations. In above examples, both knowledge graph and traffic network can be modeled as a large graph $G(V, E)$. The query of shortest path with vertex constraint can be defined as follows: given a starting vertex v_s, an ending vertex v_e, and a subset $V_s \subseteq V$, find a path with the minimum length among all the paths passing through every $v_i \in V_s$ from v_s to v_e. The subset V_s is called vertex constraint; that is, the shortest path must pass through every vertex in the subset V_s.

The above problem is a special case of *Generalized Traveling Salesman Path (GTSP)* problem [8], which is known to be NP-hard. In GTSP problem, all the vertices in G are partitioned into several categories. The objective is to find a path that visits at least one vertex for every category specified by user. For example, a tourist plans to travel through three kinds of locations, e.g., a coffee shop, a gas station, and a bank.

Because he/she may have several choices for every location category, then it is necessary to find an optimal route for him/her. The basic idea of most existing works on GTSP problem is as follows: they first compute all permutations for given categories. Each permutation represents a class of path which has the same order of the categories. Next, for every permutation, these methods enumerate all possible paths from source to destination by concatenating the subpaths between vertices in two successive categories. Finally, they find the optimal one from these paths. In our problem, every vertex in G represents a category different to others. Thus these methods need to calculate all the permutations of the vertices to be visited, which incur too heavy computational consumption. However, most of these permutations are unnecessary for computing the shortest path. Therefore, the main challenge is how to avoid computing unnecessary permutations when finding the shortest path with vertex constraint. In this paper, we propose a novel efficient algorithm based on the best-first search to compute the shortest path with vertex constraint. The main idea of our method is to avoid calculating the unnecessary permutations as soon as possible. We also propose an approximate algorithm in polynomial time which is more efficient for large graphs. The contributions of this paper are summarized below.

(i) We propose a novel and efficient exact heuristic algorithm with two optimizing techniques to find the shortest path with vertex constraint.

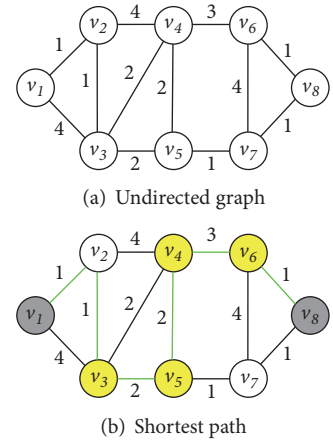
(ii) We also propose an approximate algorithm in polynomial time for our problem over large graphs. We prove the ratio bound of our approximate algorithm is 3.

(iii) We conduct extensive experiments on several real-life datasets. We compare our algorithms with the state-of-the-art methods. The experimental results validate the efficiency and effectiveness of our algorithms.

The rest of this paper is organized as follows. Section 2 gives the problem statement. Section 3 introduces the CH technique for preprocessing graphs. Section 4 proposes the best-first searching algorithm with two optimizing techniques. Section 5 proposes the approximate algorithm and analyzes the ratio bound. The experimental results are presented in Section 6. The related work is in Section 7. Finally, we conclude this paper in Section 8.

2. Problem Statement

An undirected weighted graph is denoted as $G(V, E, w)$ (or G for short), where $V = \{v_i\}$ is the set of vertices and $E \subseteq V \times V$ is the set of edges in G. w is a function that assigns a nonnegative weight $w_{i,j}$ on every edge $(v_i, v_j) \in E$; i.e., $w((v_i, v_j)) = w_{i,j}$. Note that (v_i, v_j) is equivalent to (v_j, v_i) because G is an undirected graph. The number of vertices (or edges) is denoted as $|V|$ (or $|E|$) in G. A path p in G is a sequence of vertices; i.e., $p = (v_1, v_2, \ldots, v_k)$, where every (v_i, v_{i+1}) is an edge in G for $1 \le i \le k-1$. The weight of path p, denoted as $w(p)$, is the sum of the weights of all the edges in p; i.e., $w(p) = \sum_{1 \le i \le k-1} w_{i,i+1}$. We say a path p is simple if and only if there is no repeated vertex in p. The shortest path between v_i and v_j is a path with the minimum $w(p)$ among all the paths between v_i and v_j. For simplicity, in the following, we use $w^*_{i,j}$ to denote the weight of the shortest path between v_i and v_j in G.



(a) Undirected graph

(b) Shortest path

Figure 1: An example of the shortest path with vertex constraint.

In this paper, we study the problem of finding the shortest path with vertex constraint. Table 1 summarizes the symbols in this paper. We first give the definition below.

Definition 1 (shortest path with vertex constraint). Given a graph G, a vertex subset $V_s \subseteq V$, a starting vertex v_s, and an ending vertex v_e in G, a path is called the shortest path between v_s and v_e with vertex constraint of V_s, denoted as $p^*_{s,e}$, if it satisfies the following two conditions: (1) $p^*_{s,e}$ travels through all the vertices in V_s; i.e., $v \in p^*_{s,e}$ for every vertex $v \in V_s$ and (2) $p^*_{s,e}$ is with the minimum weight among all the paths satisfying the condition (1).

Figure 1 illustrates an example of the shortest path with vertex constraint. In this example, V_s is $\{v_3, v_4, v_5, v_6\}$ and these vertices are colored with yellow in Figure 1(b). Two gray vertices, v_1 and v_8, are the starting vertex and the ending vertex, respectively. Therefore, the shortest path between v_1 and v_8 with vertex constraint of V_s is $p = (v_1, v_2, v_3, v_5, v_4, v_6, v_8)$, which is shown as the green path in Figure 1(b).

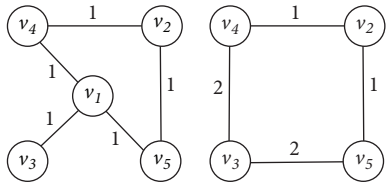
Hamilton path problem is a special case of our problem; then, we have the following theorem straightforwardly.

Theorem 2. *The problem of finding the shortest path with vertex constraint over graphs is NP-hard.*

Proof. We proof it by reducing Hamilton path problem, which is NP-complete. Given a undirected graph $G = (V, E, w)$, let v_s and v_e denote starting vertex and ending vertex, respectively. The weight of every edge in G is set as one. The vertex subset $V_s \subseteq V$ is set as $V_s = V \backslash \{v_s, v_e\}$. Obviously, there exists a Hamilton path from v_s to v_e in G if and only if the length is $|V| - 1$ for the shortest path from v_s to v_e with vertex constraint of V_s. This reduction can be done in polynomial time. Therefore, the problem of finding the shortest path with vertex constraint over graphs is NP-hard. □

Table 1: List of notations.

Symbol	Meaning
$G(V, E, w)$	An undirected weighted graph
v_s, v_e, V_s	Starting vertex, ending vertex, vertex constraint
$w_{i,j}, w(p)$	Weight of edge (v_i, v_j), weight of path p
$p^*_{s,e}$	Shortest path between v_i and v_j with vertex constraint
π	Permutation of a vertex subset
$p^*_{s,e}\|_\pi$	Shortest path between v_s and v_e under permutation π



(a) Before contraction (b) After contraction

Figure 2: Contraction of vertex.

3. CH Technique for Preprocessing Graphs

Contraction Hierarchies (CH) proposed in [9] is a well-known technique for speeding up the traditional shortest path query effectively. It essentially builds an index by maintaining the shortest paths for some pairs of vertices. In this paper, we use CH technique for preprocessing graphs to make our method more efficient.

Given a graph $G(V, E, w)$, CH first sorts all vertices in an ascending order and then contracts the vertices one by one under this order. Contraction of vertex v_i can be described as removing v_i from a graph by adding new edges which represent the shortest path between two vertices adjacent to v_i. Such edges are called shortcut edges. Specifically, for each pair of incoming edge (v_j, v_i) and outgoing edge (v_i, v_k) of v_i, if (v_j, v_i, v_k) is a unique shortest path, then a new shortcut edge (v_j, v_k) is added with weight $w_{j,i} + w_{i,k}$ to obtain a new graph G'.

We use an example in Figure 2 to illustrate the process of vertex contraction. Figure 2(a) shows a graph before the contraction of v_1. Note that there are two shortest paths between v_4 and v_5, which are (v_4, v_1, v_5) and (v_4, v_2, v_5), respectively. Thus it is unnecessary to add the edge from v_4 to v_5 when removing v_1. We also note that there is only one shortest path from v_3 to v_4. Because this path goes through v_1, a new edge from v_3 to v_4 can be constructed by removing v_1. Similarly, a new edge from v_3 to v_5 also can be constructed. Both the weights of such two new edges are 2. The result graph after contraction of v_1 is shown in Figure 2(b).

After contracting vertices, CH divides G' into an upward graph G_u and a downward graph G_d. The shortest paths can be calculated on G_u and G_d. Given a starting vertex v_s and an ending vertex v_e, a forward Dijkstra [10] search from v_s and a backward Dijkstra search from v_e are executed on G_u and G_d, respectively. The more details about CH technique are given in [9].

4. Permutation-Expanding Algorithm

In this section, we propose an algorithm to find the shortest path with vertex constraint. We first introduce the definition of permutation expanding, which is the basis of our algorithm, and then we explain the algorithm Permutation-Expanding. Two optimizing techniques are proposed in Section 4.3 and we analyze the time and space complexity of our algorithm in Section 4.4

4.1. Permutation-Expanding. Given a vertex subset V_s on G, $|V_s| = r$, a permutation π of V_s is a sequence $v_1 v_2 \cdots v_r$ of all vertices in V_s, where every $v_i \in V_s$ and $v_i \neq v_j$ for $1 \leq i, j \leq r$, $i \neq j$. Obviously, there are $r!$ permutations for a given V_s. We use $v_i \prec v_j$ to denote that if v_i is before v_j in π, a permutation is essentially an order of the vertices in V_s. We say a path p is under a permutation π, denoted as $p|_\pi$, if it satisfies the following two conditions: (1) $v_i \in p$ for every $v_i \in V_s$ and (2) there exists a subpath $v_i \rightsquigarrow v_j$ from v_i to v_j if $v_i \prec v_j$ in π. Given a $p|_\pi$, path p can be divided into $r + 1$ subpaths $v_0 \rightsquigarrow v_1, v_1 \rightsquigarrow v_2, \ldots, v_r \rightsquigarrow v_{r+1}$, where v_0 and v_{r+1} are the starting vertex and the ending vertex of p, respectively. Each $v_i \rightsquigarrow v_{i+1}$ $(0 \leq i \leq r)$ is called a "segment" of p. We use $S_{p|_\pi}$ to denote the set of all the segments of p.

In the example of Figure 1, $p = (v_1, v_2, v_3, v_5, v_4, v_6, v_8)$ is a path under permutation $\pi = v_3 v_5 v_4 v_6$. Here, $S_{p|_\pi} = \{(v_1, v_2, v_3), (v_3, v_5), (v_5, v_4), (v_4, v_6), (v_6, v_8)\}$.

A path is called the shortest path between v_s and v_e under permutation π, denoted as $p^*_{s,e}|_\pi$, if every segment $v_i \rightsquigarrow v_j \in S_{p^*_{s,e}|_\pi}$ is a shortest path. Then we have the following theorem.

Theorem 3. *Given an undirected graph G, a vertex subset V_s, a starting vertex v_s, and an ending vertex v_e in G, the shortest path $p^*_{s,e}$ between v_s and v_e with vertex constraint of V_s is exactly $p^*_{s,e}|_\pi$ with the minimum weight among all the permutations of V_s; i.e., $p^*_{s,e} = \min\{p^*_{s,e}|_\pi \mid \pi \in \Pi(V_s)\}$, where $\Pi(V_s)$ is the set of all permutations of V_s.*

Proof. Assuming that $p_{s,e}|_{\pi'}$ is a path under a permutation π' from v_s to v_e and the weight of $p_{s,e}|_{\pi'}$ is less than that of $p^*_{s,e}|_\pi$, then there will be the following four situations.

(1) If π' and π are the same permutations and every segment $v_i \rightsquigarrow v_j \in S_{p^*_{s,e}|_{\pi'}}$ is a shortest path, obviously $p_{s,e}|_{\pi'}$ and $p^*_{s,e}|_\pi$ have the same weight. This contradicts the assumption.

(2) If π' and π are the same permutations and not all of the segments $v_i \rightsquigarrow v_j \in S_{p^*_{s,e}|_{\pi'}}$ are shortest path, obviously the weight of $p_{s,e}|_{\pi'}$ is greater than that of $p^*_{s,e}|_\pi$. This contradicts the assumption.

(3) If π' and π are different permutations and every segment $v_i \rightsquigarrow v_j \in S_{p^*_{s,e}|_{\pi'}}$ is a shortest path, because $p^*_{s,e}|_\pi$ is the path with the minimum weight among all the permutations, the weight of $p^*_{s,e}|_\pi$ is less than that of $p_{s,e}|_{\pi'}$. This contradicts the assumption.

(4) If π' and π are different permutations and not all of the segments $v_i \rightsquigarrow v_j \in S_{p^*_{s,e}|_{\pi'}}$ are shortest path, obviously the weight of $p_{s,e}|_{\pi'}$ is not smaller than that of $p^*_{s,e}|_\pi$. This contradicts the assumption.

To sum up, $p^*_{s,e}|_\pi$ is the shortest path between v_s and v_e with vertex constraint of V_s. □

For two vertex subsets V_s and V'_s on G, if $V_s \subseteq V'_s$, for every permutation π of V_s, there must exist a permutation π' of V'_s, such that $v_i \prec v_j$ in π' if $v_i \prec v_j$ in π. π is a subpermutation of π', denoted as $\pi \subseteq \pi'$. If $|V_s| = r$, π is also called a r-permutation of vertex set V_s. Specifically, π is called a prefix of π', denoted as $\pi \subseteq_p \pi'$, if $\pi \subseteq \pi'$ and π is at the beginning of π'. For example, v_3v_4 is a prefix of $v_3v_4v_5v_6$.

Given a permutation π, $\pi' = \pi \oplus v$ is an *expanded permutation* with one vertex v from π, where $\oplus$ is the concatenation operator appending v at the end of π. Obviously, $\pi \subseteq_p \pi'$ and $|\pi'| = |\pi| + 1$. This process is called *permutation expanding*. For the example in Figure 1, given a permutation $\pi = v_3v_4$, $\pi' = v_3v_4v_5$ and $\pi'' = v_3v_4v_6$ are two expanded permutations with one vertex v_5 and v_6, respectively.

4.2. Main Algorithm. We propose an algorithm, Permutation-Expanding, to find the shortest path with vertex constraint by expanding permutation incrementally. The main idea of the algorithm is essentially best-first searching on the shortest paths under 1-permutation to r-permutation of V_s as soon as possible, until the optimal one has been searched.

The pseudocode of Permutation-Expanding is shown in Algorithm 1. Algorithm 1 utilizes a min priority queue Q to maintain a set of tuples $(\pi, w(\pi))$ (line 1). π is a subpermutation of V_s. $w(\pi)$ is the weight of the shortest path under π from v_s to the last vertex of π. If $\pi = v_1 \cdots v_k$, then $w(\pi) = w(p^*_{s,1}) + \sum_{i=1}^{k-1} w(p^*_{i,i+1})$. Here $p^*_{i,j}$ represents the shortest path without vertex constraint and $w(p^*_{i,j})$ can be easily calculated by CH technique as discussed in Section 3. Initially, Q only contains all the 1-permutations π of V_s with its $w(\pi)$ (lines 2-3). Algorithm 1 dequeues $(\pi, w(\pi))$ iteratively according to $w(\pi)$. In each iteration, a $(\pi, w(\pi))$ with the minimum $w(\pi)$ is dequeued from Q (line 11). Let V_π be the vertex set of π. If $V_\pi \neq V_s$, the algorithm generates every permutation π' by appending every vertex $v \in V_s - V_\pi$ at the end of π and enqueues $(\pi', w(\pi'))$ into Q. Otherwise, π is a permutation of V_s; Algorithm 1 generates $\pi \oplus v_e$ and enqueues it into Q (lines 6-10). Algorithm 1 terminates when a permutation $\pi \oplus v_e$ is dequeued for the first time, where π is a permutation of V_s (line 5). At this moment, $w(\pi \oplus v_e)$ is the weight of the shortest path $p^*_{s,e}$ with vertex constraint of V_s and we can obtain $p^*_{s,e}$ by the CH technique (line 12). There is a special case that no path is between v_s (or v_e) and v_i where $v_i \in V_s$. Algorithm 1 can find such case by computing the shortest path between two vertices. For such case, we return no solution for this problem.

Example 4. Given a graph G shown in Figure 1(a), let $v_s = v_1$, $v_e = v_8$, and $V_s = \{v_4, v_6\}$. Algorithm 1 first enqueues $(v_4, 4)$ and $(v_6, 7)$ into Q and then dequeues the first entry $(v_4, 4)$ from Q. $(v_4v_6, 7)$ is enqueued into Q. Then the entry $(v_6, 7)$ is dequeued from Q. $(v_6v_4, 10)$ is enqueued into Q. Then the entry $(v_4v_6, 7)$ is dequeued from Q. Due to $V_\pi = V_s$ where $\pi = v_4v_6$, $(v_4v_6v_8, 8)$ is enqueued into Q. Then the entry $(v_4v_6v_8, 8)$ is dequeued from Q. Due to the fact that the last vertex of π is the ending vertex v_8, where $\pi = v_4v_6v_8$, Algorithm 1 returns $p^*_{s,e} = (v_1, v_2, v_3, v_4, v_6, v_8)$ as the shortest path with vertex constraint of V_s.

4.3. Optimizing Techniques. We give two optimizing techniques to improve the efficiency of Permutation-Expanding algorithm.

Cache Mechanism. Given two different permutations π and π', there may exist the overlapping segments for the shortest paths under π and π'. The weights of these overlapping shortest subpaths are unnecessary to be calculated for many times during the permutation expanding. Cache Mechanism is utilized to maintain these values. For the example in Figure 1(a), v_1 and v_8 are the starting and ending vertices, respectively, and $V_s = \{v_3, v_4, v_5, v_6\}$ is the vertex constraint. Let $\pi = v_3v_4v_5v_6$ and $\pi' = v_6v_4v_5v_3$. Obviously, π and π' are two permutations of V_s. When calculating the shortest path between v_4 and v_5 for the first time, the distance between v_4 and v_5 is maintained and it only needs to be calculated once when π and π' are both expanded in Permutation-Expanding. The experimental results validate that Cache Mechanism can avoid redundant calculation effectively.

Permutation Filtering. When a permutation π is dequeued from Q in an iteration, Permutation-Expanding generates all expanded permutations $\pi' = \pi \oplus v$ by appending every vertex $v \in V_s - V_\pi$ at the end of π. Note that it is unnecessary to enqueue every π' into Q in this iteration. For two expanded permutations $\pi'_i = \pi \oplus v_i$ and $\pi'_j = \pi \oplus v_j$, $v_i \neq v_j$, if the shortest path $p^*_{s,i}$ under π'_i between v_s and v_i is a subpath of $p^*_{s,j}$, then permutation π'_j can be filtered and it does not need to be enqueued into Q. The following theorem guarantees the correctness of permutation filtering.

Theorem 5. *For two expanded permutations $\pi'_i = \pi \oplus v_i$ and $\pi'_j = \pi \oplus v_j$, $v_i \neq v_j$, if the shortest path $p^*_{s,i}$ under π'_i from v_s to v_i is a subpath of $p^*_{s,j}$, then for any permutation π''_j of V_s, $\pi'_j \subseteq_p \pi''_j$, there exists a permutation π''_i of V_s, $\pi'_i \subseteq_p \pi''_i$, such that the weight of the shortest path under π''_j from v_s to v_e must*

Input: G, V_s, v_s, v_e.
Output: $p^*_{s,e}$.
// Input: G: an undirected weighted graph
// V_s: a vertex subset of V
// v_s, v_e: starting vertex and ending vertex respectively
// Output: $p^*_{s,e}$: the shortest path between v_s and v_e with vertex
// constraint of V_s
1: Let Q be a min priority queue with entries in the form $(\pi, w(\pi))$, sorted in ascending order of $w(\pi)$;
2: **for** each $v_j \in V_s$ **do**
3: Enqueue an entry $(v_j, w(p^*_{s,j}))$ into Q;
4: Dequeue the first entry $(\pi, w(\pi))$ from Q and let v_i be the last vertex of π;
5: **while** $v_i \neq v_e$ **do**
6: **if** $V_\pi \neq V_s$ **then**
7: **for** each $v_j \in V_s - V_\pi$ **do**
8: Enqueue an entry $(\pi \oplus v_j, w(\pi) + w(p^*_{i,j}))$ into Q;
9: **else**
10: Enqueue an entry $(\pi \oplus v_e, w(\pi) + w(p^*_{i,e}))$ into Q;
11: Dequeue the first entry $(\pi, w(\pi))$ from Q and let v_i be the last vertex of π;
12: Generate the shortest path $p^*_{s,e}$ between v_s and v_e under a permutation π;
13: **return** $p^*_{s,e}$;

Algorithm 1: Permutation-Expanding (G, V_s, v_s, v_e).

not be less than the weight of the shortest path under π''_i from v_s to v_e.

Proof. Given a shortest path p^* under π''_j from v_s to v_e, $p^*_{s,j}$ is obvious a prefix subpath of p^*. Let $p^*_{i,j}$ denote the shortest path from v_i to v_j. Consider the path $p^*_{s,i} \oplus p^*_{i,j}$ obtained by concatenating $p^*_{s,i}$ and $p^*_{i,j}$, because $p^*_{s,i}$ is a subpath of $p^*_{s,j}$, then we have $w(p^*_{s,i} \oplus p^*_{i,j}) \leq w(p^*_{s,j})$, where $w(p)$ represents the weight of path p. Next, we consider the subpath $v_j \rightsquigarrow v_e$ of p^*; $v_j \rightsquigarrow v_e$ must go through v_i. Let v_i^- and v_i^+ represent the precursor and successor of v_i in subpath $v_j \rightsquigarrow v_e$. A new path $p^*_{j,e}$ can be obtained by utilizing the shortest path from v_i^- to v_i^+ to replace the part $v_i^- \longrightarrow v_i \longrightarrow v_i^+$ in $v_j \rightsquigarrow v_e$. It is obvious that $w(p^*_{j,e}) \leq w(v_j \rightsquigarrow v_e)$. We concatenate $p^*_{s,i}$, $p^*_{i,j}$, and $p^*_{j,e}$ to get a path p' from v_s to v_e. Obviously, p' is a path under a permutation π''_i of V_s and we have $w(p') \leq w(p^*)$. Theorem 5 has been proved. □

The conclusion of Theorem 5 is obvious. For the example in Figure 1, let $v_s = v_1$, $v_e = v_8$, and $V_s = \{v_4, v_6\}$. We consider two permutations v_1v_4 and v_1v_6, which are expanded from v_1. The shortest paths under v_1v_4 and v_1v_6 are $p = (v_1, v_2, v_3, v_4)$ and $p' = (v_1, v_2, v_3, v_4, v_6)$, respectively. Because p is a subpath of p', v_1v_6 does not need to be enqueued into Q in the iteration when $\pi = v_1$ is dequeued from Q. The reason is that all the paths under the permutations expanded from v_1v_6 cannot be the shortest path with vertex constraint.

4.4. Complexity Analysis. In this section, we analyze the complexity of Algorithm 1. We first analyze the time complexity and then analyze the space complexity.

Time Complexity. Because Algorithm 1 may calculate the shortest path for every two vertices in V_s in the worst case, it needs at most $(r + 1)(r + 2)$ calculations for the shortest paths, where $r = |V_s|$. For each shortest path calculation, CH runs in $O(n \log n + m)$ time where $n = |V|$ and $m = |E|$. In addition, at most $r!$ permutations of V_s may be created and every permutation is maintained as a tuple which can be done in O(1) time. Therefore, Algorithm 1 runs in $O(r^2(n \log n + m) + r!)$ time. It is worth noting that r is always far less than n in real applications.

Space Complexity. Algorithm 1 mainly needs to maintain the expanded permutations and expand at most $r!$ permutations. Therefore, the space complexity of Algorithm 1 is $O(r!)$.

5. Approximate-Path Algorithm

In this section, we propose an approximate algorithm Approximate-Path to find the shortest path with vertex constraint in polynomial time. In the following, we first define query graph and then explain our approximate algorithm in detail. Next, we prove that the ratio bound of our approximate algorithm is 3. Finally, we analyze the time and space complexity of Approximate-Path.

Given a graph G, a vertex subset $V_s \subseteq V$, a starting vertex v_s, and an ending vertex v_e in G, a query graph $G_q(V_q, E_q)$ is a complete graph on V_q, where $V_q = V_s \cup \{v_s, v_e\}$, $E_q = \{(v_i, v_j) \mid v_i, v_j \in V_q, v_i \neq v_j\}$. In G_q, the weight $w_{i,j}$ of every edge (v_i, v_j) is the shortest distance $w^*_{i,j}$ between v_i and v_j in G. Here, $w^*_{i,j}$ is weight of the shortest path between v_i and v_j without vertex constraint in G.

The following theorem indicates that we only need to find the shortest path with vertex constraint over G_q.

Input: G, V_s, v_s, v_e.
Output: p.
// Input: G: an undirected weighted graph
// V_s: a vertex subset of V
// v_s, v_e: starting vertex and ending vertex respectively
// Output: p: the approximate shortest path between v_s and v_e
// with vertexconstraint of V_s
1: Let Q be a min priority queue with the entries in the form $\langle v_i, v_j, w^*_{i,j}\rangle$, sorted in the ascending order of $w^*_{i,j}$, where $w^*_{i,j}$ is the shortest distance between v_i and v_j;
2: $V_q \longleftarrow V_s \cup \{v_s, v_e\}, m \longleftarrow |V_q|$;
3: **for** each $v_k \in V_q - \{v_s\}$ **do**
4: Enqueue an entry $\langle v_s, v_k, w^*_{s,k}\rangle$ into Q;
5: $E_T \longleftarrow \emptyset, V_T \longleftarrow \{v_s\}$;
6: **while** $V_T \neq V_q$ **do**
7: Dequeue the first entry $\langle v_i, v_j, w^*_{i,j}\rangle$ from Q;
8: **if** $v_j \in V_T$ **then**
9: **continue**;
10: **else**
11: $V_T \longleftarrow V_T \cup \{v_j\}, E_T = E_T \cup \{(v_i, v_j)\}$;
12: **for** each $v_k \in V_q - V_T$ **do**
13: Enqueue an entry $\langle v_j, v_k, w^*_{j,k}\rangle$ into Q;
14: $T \longleftarrow (V_T, E_T)$;
15: Traverse T by preorder and let $\pi = v_1 v_2 \cdots v_m$ be a permutation corresponding to the order of vertices in preorder traversal on T;
16: Move the ending vertex v_e to the end of π to get $\pi' = v'_1 v'_2 \cdots v'_m$;
17: Generate the shortest path p between v_s and v_e under a permutation π';
18: **return** p;

Algorithm 2: Approximate-Path (G, V_s, v_s, v_e).

Theorem 6. *It is identical for the weight of the shortest path between v_s and v_e with vertex constraint of V_s in G and G_q.*

The main idea of Approximate-Path is as follows. We first compute the minimum spanning tree T of G_q and then "adjust" some edges in T such that T is converted into a path p satisfying the vertex constraint. The pseudocode of Approximate-Path is shown in Algorithm 2. In Algorithm 2, the minimum spanning tree T of G_q is first generated in a similar way as *Prim Algorithm* [11] (lines 1-14). Next, Algorithm 2 executes a preorder traversal on T and then we have a permutation π corresponding to the order of vertices in such preorder traversal on T (line 15). Note that in π the ending vertex v_e may not be the last one. In this case, v_e is put into the end of π and we get a new permutation π' (line 16). Finally, Algorithm 2 returns the shortest path under permutation π' as a result (lines 17-18), which is an approximate solution for our problem.

Example 7. Figures 3(a) and 3(b) show the query graph G_q and the minimum spanning tree T of G_q, respectively. Let $\pi = v_1 v_3 v_4 v_5 v_8 v_6$ be a permutation corresponding to the preorder traversal on T shown in Figure 3(c). Then Approximate-Path removes the ending vertex v_8 to the end of π to get $\pi' = v_1 v_3 v_4 v_5 v_6 v_8$. The path between v_1 and v_8 under π' in G_q is $(v_1, v_3, v_4, v_5, v_6, v_8)$ shown in Figure 3(d), and its weight is 12. The shortest path with vertex constraint V_s for the input graph is shown in Figure 1(b) and its weight is 10.

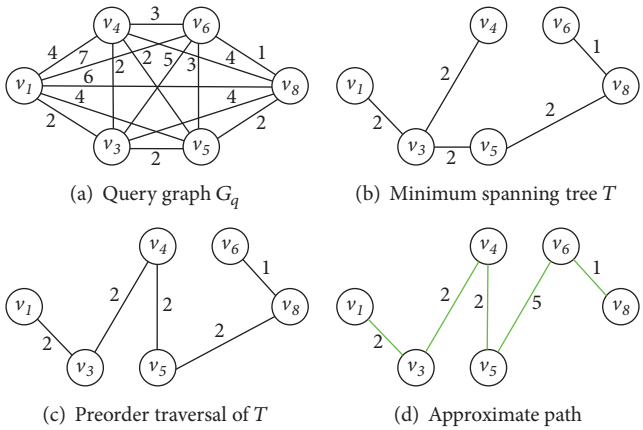


Figure 3: An example of an approximate path.

Next, we prove that Approximate-Path is a 3-approximation algorithm for shortest path problem with vertex constraint.

Theorem 8. *Approximate-Path is a 3-approximation algorithm for finding the shortest path with vertex constraint.*

Proof. Let $p^*_{s,e}$ denote a shortest path with vertex constraint of V_s in G_q. Obviously, $p^*_{s,e}$ is a spanning tree of G_q. Therefore, the weight of the minimum spanning tree T of G_q, computed

by Approximate-Path, provides a lower bound on the weight of $p^*_{s,e}$:

$$w(T) \leq w\left(p^*_{s,e}\right) \tag{1}$$

The preorder traversal π of T is essentially a vertex permutation of V_q. Let $\pi = v_1 v_2 \cdots v_{|V_q|}$, where $v_i \in V_q$ for $1 \leq i \leq |V_q|$. We use $p|_{T,\pi}$ to denote a path on T under permutation π. Note that $p|_{T,\pi}$ may not be a simple path and every edge in $p|_{T,\pi}$ appears at most twice. For the example in Figure 3, $\pi = v_1 v_3 v_4 v_5 v_8 v_6$ and its $p|_{T,\pi} = (v_1, v_3, v_4, v_3, v_5, v_8, v_6)$. Here, the edge (v_3, v_4) (or (v_4, v_3)) appears twice in $p|_{T,\pi}$. Because $p|_{T,\pi}$ travels through every edge in T at most twice, then we have

$$w\left(p|_{T,\pi}\right) \leq 2w(T) \tag{2}$$

Based on inequality (1) and equation (2), we have

$$w\left(p|_{T,\pi}\right) \leq 2w\left(p^*_{s,e}\right) \tag{3}$$

Because G_q is a complete graph, we can generate a simple path $p|_{G_q,\pi}$ on G_q under permutation π. Note that if $\pi = v_1 v_2 \cdots v_{|V_q|}$, then $p|_{G_q,\pi} = (v_1, v_2, \ldots, v_{|V_q|})$ and every (v_i, v_{i+1}) is an edge in G_q for $1 \leq i \leq |V_q| - 1$. Additionally, the weight of every edge (v_i, v_j) in G_q is equal to the weight of the shortest path between v_i and v_j in G; thus, the weight of edge (v_i, v_j) cannot be larger than the weight of subpath between v_i and v_j in $p|_{T,\pi}$. It means

$$w\left(p|_{G_q,\pi}\right) \leq w\left(p|_{T,\pi}\right) \tag{4}$$

Given the permutation π of preorder traversal of T, Algorithm 2 obtains another permutation π' by removing the ending vertex v_e to the end of π. For the last two vertices $v_{|V_q|}$ and v_e of π', if $(v_{|V_q|}, v_e)$ is an edge in T, its weight must be less than the weight of T. Otherwise, there must exist a simple path between $v_{|V_q|}$ and v_e in T and its weight cannot be less than the shortest distance between $v_{|V_q|}$ and v_e. Therefore, for both two cases, $w^*_{|V_q|,e} \leq w(T)$ and then we have

$$\begin{aligned} w\left(p|_{G_q,\pi'}\right) &\leq w\left(p|_{G_q,\pi}\right) + w(T) \leq 3w(T) \\ &\leq 3w\left(p^*_{s,e}\right) \end{aligned} \tag{5}$$

Because $w(p|_{G_q,\pi'})$ is exactly the weight of the approximate shortest path returned by Algorithm 2, then the proof is completed. □

Complexity Analysis. We first analyze the time complexity for Algorithm 2. In order to construct the minimum spanning tree of G_q, we utilize the CH technique to calculate the weight of shortest path between any two vertices in V_s. It needs $O(r^2(n \log n + m))$ time, where $n = |V|$, $m = |E|$, and $r = |V_s|$, then the time complexity of Algorithm 2 is $O(r^2(n \log n + m))$. In order to construct the minimum spanning tree, Algorithm 2 needs to maintain the weight of shortest path for any two vertices in V_s, then the space complexity of Algorithm 2 is $O(r^2)$.

6. Experiments

This section experimentally evaluates our algorithms against the current state-of-the-art methods. Section 6.1 explains the experimental settings. Section 6.2 presents the performance of algorithms.

6.1. Experimental Settings. All methods are implemented in C++ and tested on a Linux machine with an Intel(R) Core(TM) i7-4770K and 32GB RAM. We repeat each experiment 100 times and report the average result. If a method requires more than 24 hours or more than 32GB RAM to preprocess a dataset D, we omit the method from the experiments on D.

Datasets. We test 4 real road networks from the 9th DIMACS Implementation Challenge (http://www.dis.uniroma1.it/challenge9/index.shtml) and an email network (http://snap.stanford.edu/data/) as shown in Table 2. For each graph, each vertex represents a road junction and each edge represents a road segment. Table 2 describes the properties of the datasets, where $|V|$, $|E|$, and d are the number of vertices, the number of edges in the road network, and the average degree of vertex, respectively. The full name of each road network is shown in description.

Query Set. In this paper, we investigate the query efficiency by varying the size of the vertex constraint. The size of the vertex constraint is the number of vertices in V_s. We test 15 kinds of query sets Q1 to Q15, where every query set is a set of queries with an appropriate size of V_s. For each query set, we test 100 random queries and report the average querying time and space consumption as the results for the current query set. Specifically, the sizes of V_s for Q1-Q5 are 4,5,6,7,8, respectively, and the sizes of V_s for Q6-Q10 are 12,14,16,18,20, respectively. The starting and ending vertex for every query are additionally selected in random way. Q11-Q15 are generated as follows. We first randomly select 500 pairs of the starting vertex v_s and the ending vertex v_e and then calculate distance for every pair of v_s and v_e. We sort these distances in ascending order and generate Q11-Q15 by dividing these pairs of v_s and v_e into five query sets. For example, Q11 represents the queries for the pairs of v_s and v_e whose distances are in the top 100, and so on. For each query, we randomly select six vertices as V_s; that is, the size of V_s is 6.

For a query, if the starting vertex and ending vertex are the same, we call this starting-to-starting query (STS query); otherwise, we call this starting-to-ending query (STE query). In this paper, we present the experimental results of our algorithms for both STS query and STE query.

Compared Methods. For each experiment, we compare Permutation-Expanding (PE) and Approximate-Path (AP) against three algorithms which are unidirectional Dijkstra Search (U.Dijkstra) [8], Level-Sweeping Search (LESS) [8], and Nearest Neighbor Algorithm (ANN) [12]. We use CH technique to preprocess the input graphs. The first two

Table 2: Datasets.

Dataset	$\|V\|$	$\|E\|$	d	Description
NY	264,346	733,846	2.78	New York City Road Network
BAY	321,270	800,172	2.49	San Francisco Bay Area Road Network
COL	435,666	1,057,066	2.43	Colorado Road Network
FLA	1,070,376	2,712,798	2.53	Florida Road Network
EMAIL	265,214	420,045	1.25	Email network of EU research institution

compared algorithms are exact algorithms and the last one is an approximation algorithm. The other methods are not included in our comparison for the following reasons: (1) INC [13] computes a simple path which does not contain repeated vertex; however, we do not require a simple path in this problem and (2) P-LESS [8] is an optimization algorithm of LESS and mainly achieves the size of search space which typically grows in size proportional to the density of category. When each category contains only one vertex, P-LESS is equivalent to LESS.

6.2. Experimental Results

Exp-1. Query Efficiency. We investigate the impact of the size of V_s and show the experimental results of STE query in Figure 4(a). On each dataset, we find that U.Dijkstra has the largest querying time for every query. PE outperforms LESS by large margins depending on the size of V_s for each dataset and their maximum difference is close to two orders of magnitude. The reason is that LESS calculates all the permutations of V_s. In contrast, PE finds the shortest path with vertex constraint by expanding permutation incrementally, which can avoid calculating the unnecessary permutations as soon as possible. We can see that PE begins to degrade as the size of graph increases. Despite this degradation, it only requires no more than 3 seconds in the worst case (for Q5 on FLA).

For each dataset, we find that AP has the minimum time cost than the other algorithms on every query. Specifically, AP outperforms ANN by one order of magnitude. When the size of V_s is small, our exact algorithm PE runs less time than the approximate algorithm ANN, and AP answers these queries in subsecond time. We find the querying times of ANN and AP are nonsensitive to the size of V_s in Figure 4(a).

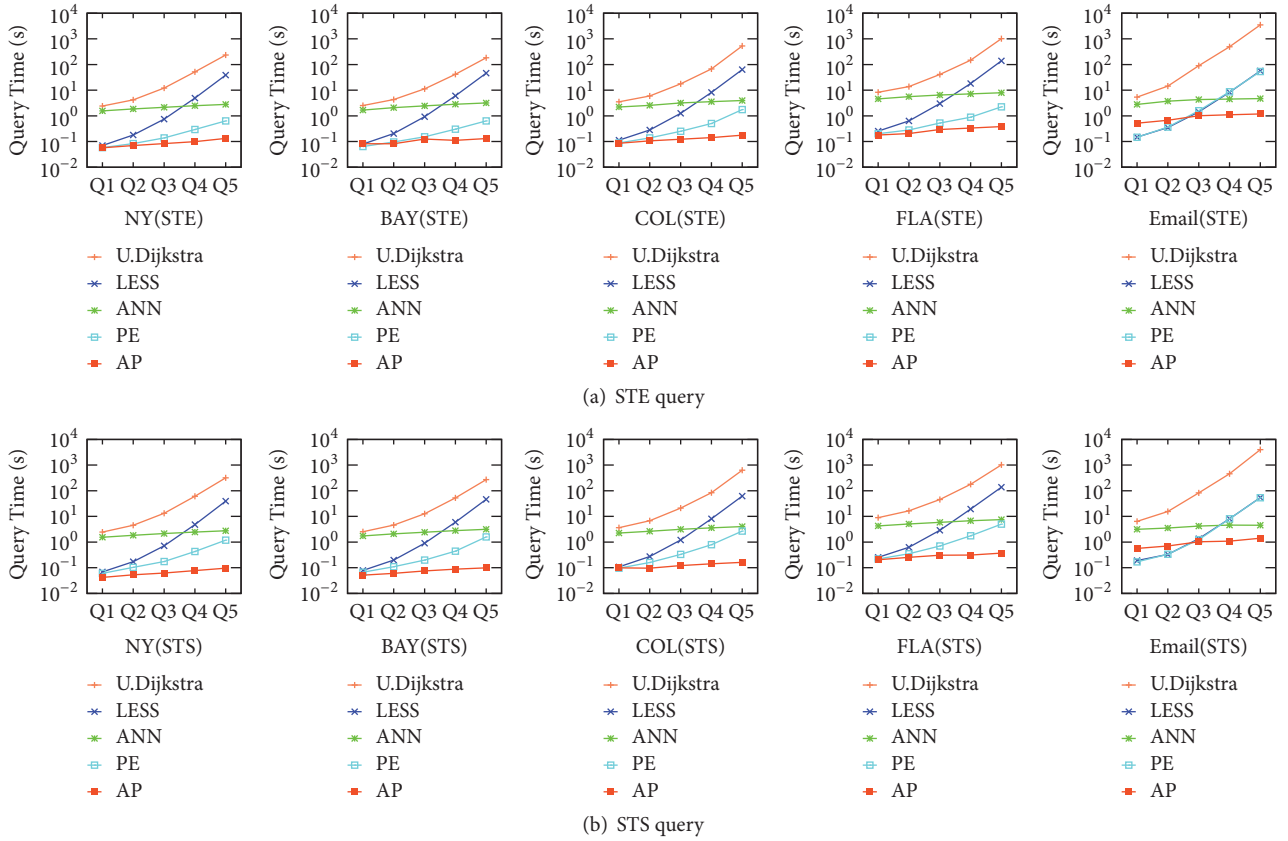
As shown in Figure 4(b), the query efficiency of STS query is similar to STE query. PE is better than the other exact algorithms and AP has the minimum time cost than the other algorithms on every query. For the same size of V_s and dataset, the querying time of STE query is less than that of STS query. The reason is that given a starting vertex, PE uses best-first searching on the shortest paths under 1-permutation to r-permutation of V_s as soon as possible, until the optimal one has been searched out. PE gradually expands the path, and finally each vertex in V_s will be arranged according to its shortest distance from the starting vertex. However, STS query eventually returns to the starting vertex, so it will generate more permutations than STE query, which increases the running time of the algorithm.

When the size of V_s becomes large, for Q6-Q10 query, because the runtime of the exact algorithms is too long, here we only compare the query efficiency of the approximate algorithms. Figure 5 shows the results of these queries. We find the performance of AP is also better than ANN by an order of magnitude and the querying time of AP does not exceed 2 seconds in the worst case for both STE query and STS query.

Q11-Q15 has the same size of V_s and the query time is shown in Figure 6. As the distance between the starting vertex and the ending vertex increases, the time required for the query does not increase. This shows that the time required for the query is not related to the distance between the starting vertex and the ending vertex but is only related to the size of V_s and the scale of the graph. For PE and AP algorithms, they find the shortest path with vertex constraint by expanding permutation incrementally, which can avoid calculating the unnecessary permutations as soon as possible. Moreover, AP can quickly give a solution to the problem by using the query graph. Therefore, AP and PE are more efficient than the other algorithms.

Figure 7 shows the space consumption of our algorithms on Q1-Q5. We can find that the space consumptions of STE query and STS query are nearly the same on every dataset. For every dataset, U.Dijkstra has the largest space consumption. PE has the smallest space consumption among all the exact algorithms and ANN has the smallest space consumption among all algorithms. Because ANN only needs to calculate the $|V_s| + 1$ shortest subpaths and does not save any intermediate calculation results, it has less space consumption than AP. Note that our approximation algorithm is with the least space consumption except ANN.

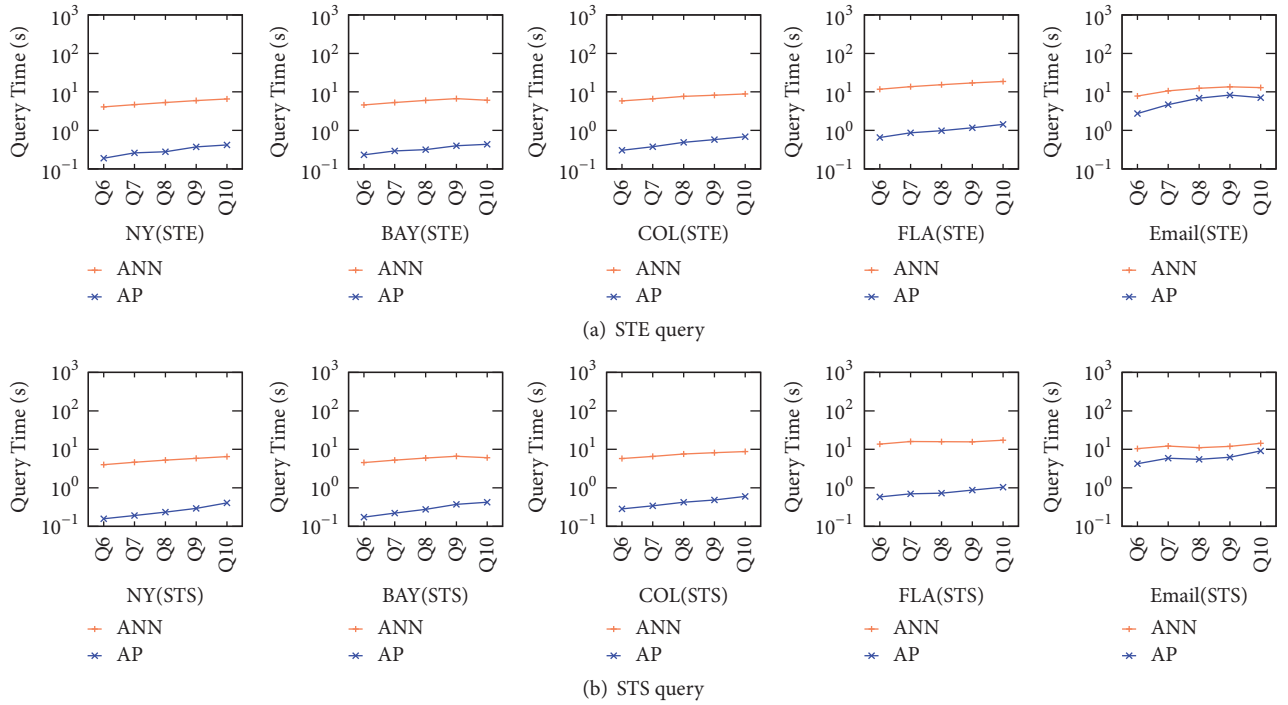
Exp-2. Effectiveness of Optimizing Techniques. For PE, we design two optimizing techniques. The optimizing effectiveness of PE is shown in Figure 8. The speedup ratio is the ratio of the query times of using optimizing techniques and without optimizing techniques. We can see that the optimizing techniques can greatly reduce the query time. Figure 8(a) shows the effectiveness of optimizing techniques on STE query. The results show that the efficiency of PE can be increased by several times through optimizing techniques depending on the size of V_s for each dataset. In addition to COL, with the increase of the size of V_s, the ratio of speedup is also increasing. For COL, due to its larger diameter but



(a) STE query

(b) STS query

Figure 4: Query efficiency on Q1-Q5.



(a) STE query

(b) STS query

Figure 5: Query efficiency on Q6-Q10.

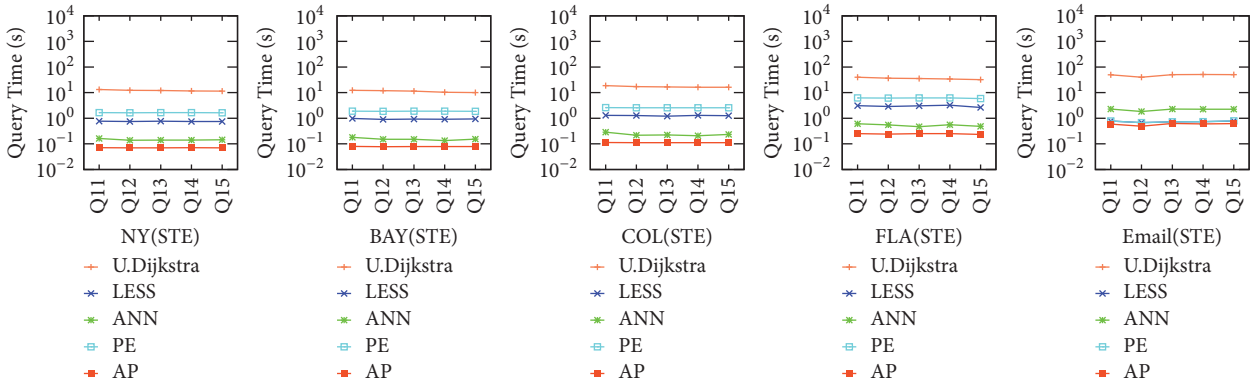
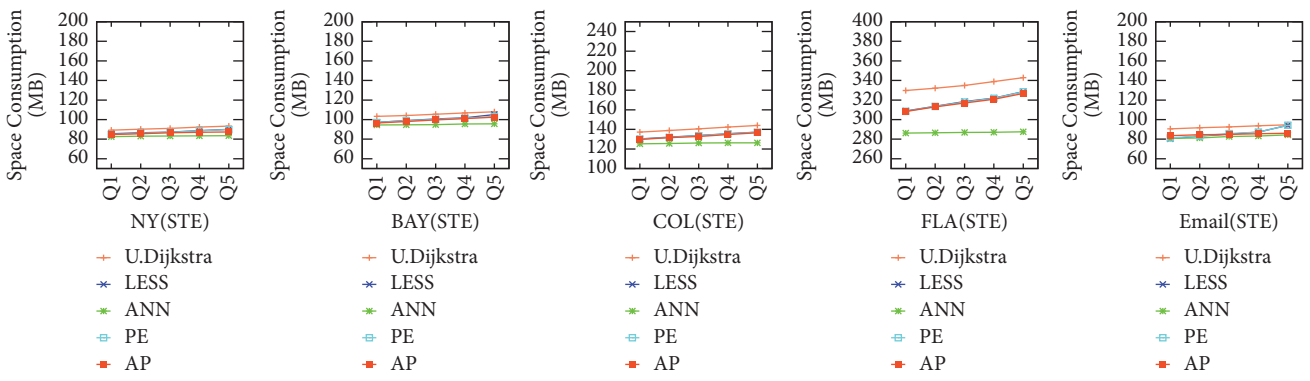
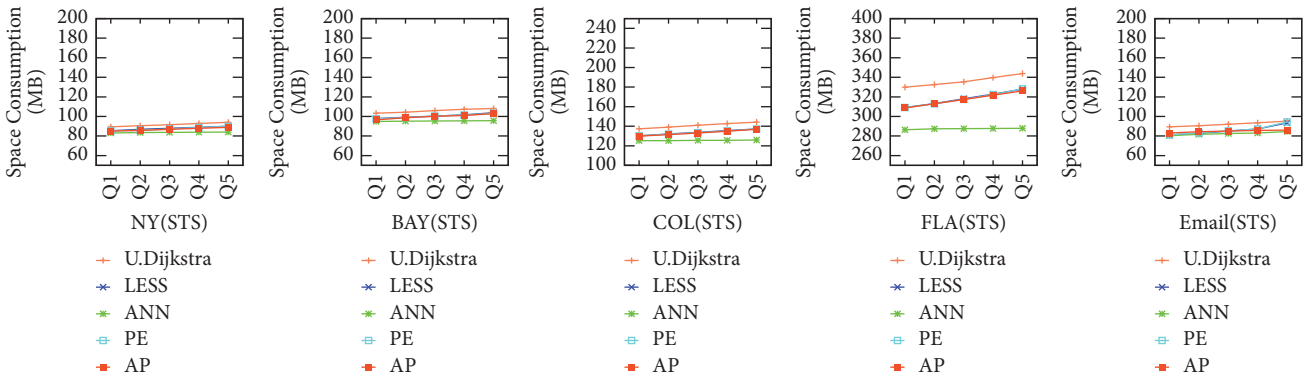


Figure 6: Query efficiency on Q11-Q15.



(a) STE query

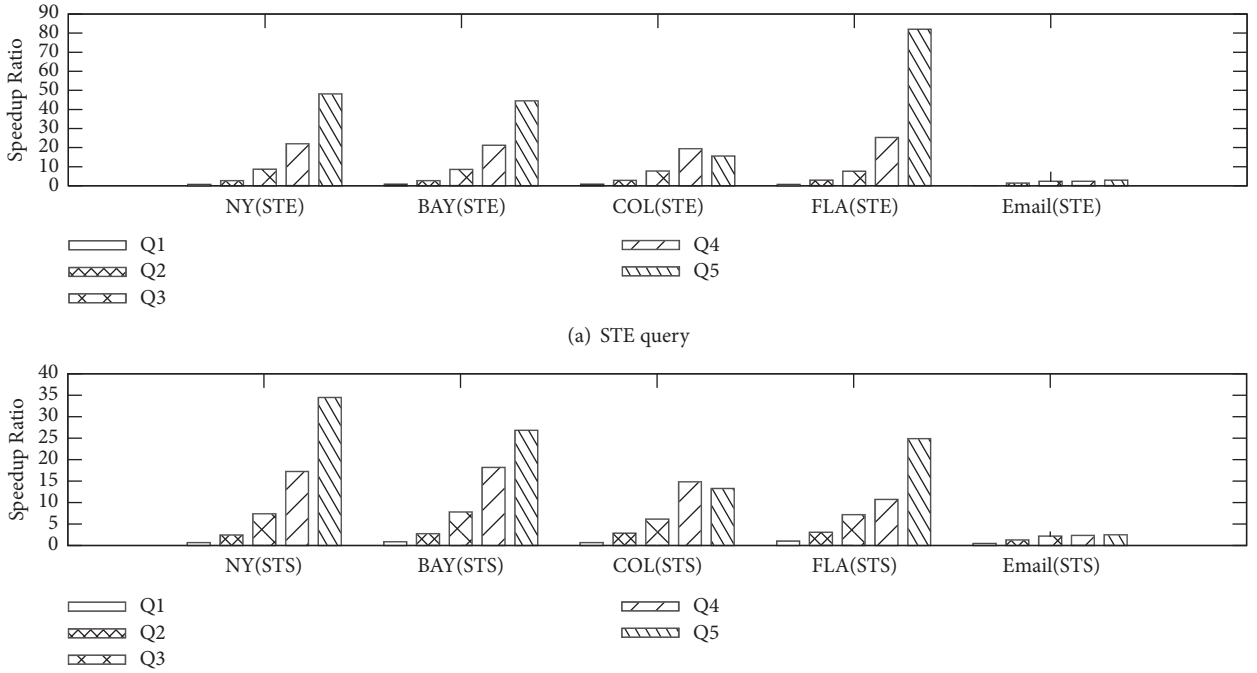


(b) STS query

Figure 7: Space consumption on Q1-Q5.

narrower width, which means that the traffic network is in strip sharp, PE can have better performance even without any optimizing technique. Consider an extreme case, when the network degenerates into a line, PE also can achieve the best performance without any optimizing technique. Of course, this kind of network is very rare in real life. Figure 8(b) shows the ratio of speedup on STS query. Since STS query needs to calculate more permutations than STE query, the ratio of speedup on STS query is relatively small.

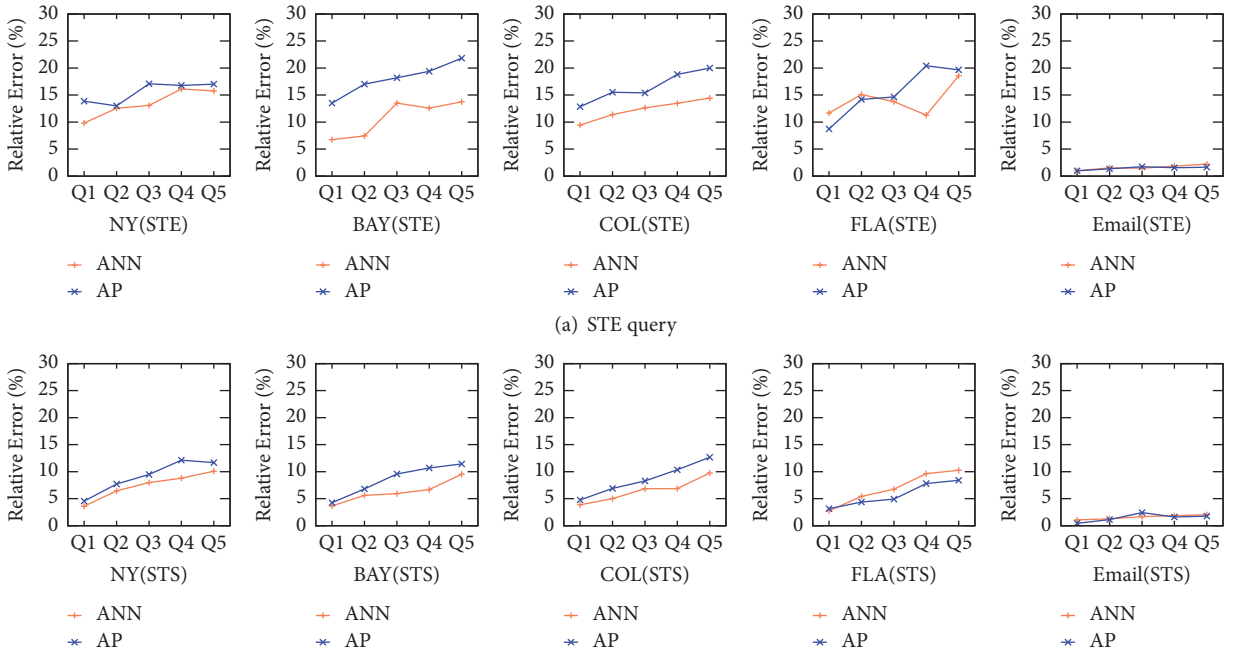
Exp-3. Relative Error. The relative error is $(W_a - W_o)/W_o$, where W_a and W_o are the weights of approximation solution and optimal solution, respectively. For every query in this group of experiments, we first use PE to calculate the optimal result, and then use ANN and AP to calculate the approximate result. Figure 9 shows the relative errors of those two approximation algorithms on the different datasets. For STE query, the relative errors in the two datasets NY and FLA are not much different. For datasets BAY and COL, the relative errors of ANN are lower than that of AP. With the increasing of the size of V_s, the relative errors of both algorithms gradually increase. In all datasets, the relative errors of AP do not exceed 25%. However, for STS query, the relative error is relatively smaller than STE query and the



(a) STE query

(b) STS query

Figure 8: Optimizing effectiveness.



(a) STE query

(b) STS query

Figure 9: Relative error on Q1-Q5.

relative errors of AP do not exceed 15%. For dataset FLA, the relative errors of AP are lower than that of ANN.

7. Related Work

In this section, we introduce existing works and categorize them as follows.

Traveling Salesman Problem (TSP). The traveling salesman problem is a very classic graph theory problem. So far, there are many algorithms to solve this problem, including exact and approximate algorithms [14]. TSP can be transformed into a linear programming problem and solved by some methods for solving linear programming [15–17]. Dorigo [18] solves TSP problem using ant colony algorithm. In this work,

ants of the artificial colony generate pheromones on the edges of the graph. As the pheromone accumulates, the path formed by the pheromone trail produces a shorter feasible solution of TSP. As time progresses, the amount of pheromone in the shorter path gradually increases. The shorter the path, the more the pheromone deposited on it. There are also some approximate algorithms that can quickly give a better solution to the TSP problem [19–21]. However, TSP is a special case of the problem we studied in this paper. All the methods for TSP cannot solve our problem when $V_s \neq V$. Additionally, these methods cannot be used for large graphs.

Generalized Traveling Salesman Problem (GTSP). The Generalized Traveling Salesman Problem is a variant of the classical Traveling Salesman Problem. It was first introduced in the late 1960s [22]. There are some exact algorithms to solve the GTSP [23–25]. Specifically, a salesman travels in n cities (each city can only be visited for one time) and has to eventually return to the starting city. Under the conditions that the distances between n cities are given and the traveling route meets certain constraints (for example, if a salesman would like to visit city 1, he/she must ensure that he/she has visited city 2 and city 3), an optimal traveling route can be explored known as *Traveling Salesman Problem with Precedence Constraint (TSPPC)*. Ascheuer et al. [26] proposes an algorithm based on branch cut to solve the asymmetric traveling salesman problem with constraints. Moon et al. [27] and Wang et al. [28] solve the traveling salesman problem with constraints by genetic algorithm and integer programming, respectively. The Hamiltonian path problem with precedence constraints is also known as the sequential ordering problem, which can be described as finding the shortest path between the specified starting point and the specified ending point, which passes through every point once and satisfies the sequence constraints. Karan et al. [29] proposes an algorithm based on the branch boundary method to solve the sequential ordering problem. The existing algorithms for solving GTSP are essentially exhaustive for each possible path and cannot be applied to large graphs. Our algorithm can be applied to large graphs very well.

Trip Planning Query (TPQ). All vertices in a graph are divided into groups, each representing a category. Trip Planning Query is to find a minimum-cost route where, for each given category, at least one vertex should be contained. Li et al. [12] introduce four algorithms for answering TPQ; these algorithms achieve various approximation ratios with respect to m and ρ. m is the size of categories and ρ is the maximum cardinality of any category. Our algorithm is a 3-approximation algorithm and the ratio bound is lower than that of the algorithm in [12]. Rice et al. [8] present two exact algorithms to solve this problem. These algorithms use an exhaustive way to search for the optimal path, which adds a lot of unnecessary calculations and greatly increases the running time of the algorithms. Hars et al. [13] propose a heuristic algorithm that follows the divide-and-conquer approach to compute a simple path which passes through all vertices specified by user. The original question is divided into two subquestions and the algorithm consists of two main steps: (1) for a given set of must-visited vertices and the corresponding visited order, consider each pair of consecutive vertices represent a subpath of the entire end-to-end path, and then calculate all candidate subpaths; (2) concatenate candidate subpaths, one from each pair of consecutive vertices, in order to establish a simple path from starting vertex to ending vertex. Since the path we are finding does not require a simple path, the algorithm does not apply to our problem. Cao et al. [30] introduce some algorithms for solving *Keyword-aware Optimal Route (KOR)* queries. A KOR query adds a cost constraint based on the category constraint,; that is, the optimal path returned should satisfy the user-specified cost budget. Shang et al. [31] propose and study a novel problem for dynamically monitoring the shortest path in spatial network, with the aim of accelerating the shortest path computation in a dynamic spatial network. Shang et al. [32] design an exact algorithm and an approximation algorithm to solve Collective Travel Planning query problem. The query finds the lowest cost route connecting multiple sources and a destination with up to k meeting points.

8. Conclusion

To find the shortest path with vertex constraint, we propose an exact algorithm named Permutation-Expanding and give two optimizing techniques to improve its efficiency. Moreover, we also propose an approximate algorithm named Approximate-Path in polynomial time for this problem over large graphs. We conduct extensive experiments on real-life datasets and compare our algorithms with the state-of-the-art methods. The experimental results validate that our algorithms always outperform the existing methods even though the size of graph or given set of vertices is large. In the future work, we will study the index techniques to facilitate the queries such that our algorithms are more time and space efficient on the larger graphs.

Data Availability

The road network datasets used to support the findings of this study are included within the article. They can be downloaded from http://www.dis.uniroma1.it/challenge9/index.shtml.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the grants of the National Natural Science Foundation of China nos. 61402323, 61572353, and U1736103, the Opening Project of State Key Laboratory of Digital Publishing Technology, and the Australian Research Council Discovery Grant DP130103051.

References

[1] Z. Tan, X. Zhao, Y. Fang, and W. Xiao, "GTrans: generic knowledge graph embedding via multi-state entities and dynamic relation spaces," *IEEE Access*, vol. 6, no. 99, pp. 8232–8244, 2018.

[2] W. Yuan, K. He, D. Guan, and G. Han, "Edge-dual graph preserving sign prediction for signed social networks," *IEEE Access*, vol. PP, no. 99, pp. 1–1, 2017.

[3] J. Zhang, H. Yang, H. Song, and Y. Zhang, "An improved archaeology algorithm based on integrated multi-source biological information for yeast protein interaction network," *IEEE Access*, vol. 5, no. 99, pp. 15893–15900, 2017.

[4] Y.-C. Chen and C. Lee, "Skyline path queries with aggregate attributes," *IEEE Access*, vol. 4, pp. 4690–4706, 2016.

[5] Y. Yuan, L. Chen, and G. Wang, "Efficiently answering probability threshold-based shortest path queries over uncertain graphs," *International Conference on Database Systems for Advanced Applications*, pp. 155–170, 2010.

[6] Y. Yuan, X. Lian, L. Chen, Y. Sun, and G. Wang, "RSkNN: kNN search on road networks by incorporating social influence," *IEEE Transactions on Knowledge & Data Engineering*, vol. 28, no. 6, pp. 1575–1588, 2016.

[7] X. Cao, L. Chen, C. Gao, and X. Xiao, "Keyword-aware optimal route search," *Proceedings of the Vldb Endowment*, vol. 5, no. 11, pp. 1136–1147, 2012.

[8] M. N. Rice and V. J. Tsotras, "Exact graph search algorithms for generalized traveling salesman path problems," in *International Symposium on Experimental Algorithms*, pp. 344–355, Springer, 2012.

[9] R. Geisberger, P. Sanders, D. Schultes, and D. Delling, "Contraction hierarchies: Faster and simpler hierarchical routing in road networks," in *International Workshop on Experimental and Efficient Algorithms*, pp. 319–333, Springer, 2008.

[10] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[11] R. C. Prim, "Shortest connection networks and some generalizations," *Bell Labs Technical Journal*, vol. 36, no. 6, pp. 1389–1401, 1957.

[12] F. Li, D. Cheng, M. Hadjieleftheriou, G. Kollios, and S. Teng, "On trip planning queries in spatial databases," in *International Symposium on Spatial and Temporal Databases*, pp. 273–290, Springer, 2005.

[13] H. Vardhan, S. Billenahalli, W. Huang et al., "Finding a simple path with multiple must-include nodes," in *Modeling, Analysis & Simulation of Computer and Telecommunication Systems MASCOTS'09*, pp. 1–3, IEEE International Symposium on. IEEE, 2009.

[14] Y. Lu, U. Benlic, and Q. Wu, "A hybrid dynamic programming and memetic algorithm to the traveling salesman problem with hotel selection," *Computers & Operations Research*, vol. 90, pp. 193–207, 2018.

[15] C. Chekuri and K. Quanrud, *Fast Approximations for Metric-Tsp via Linear Programming*, 2018.

[16] M. Diaby and M. H. Karwan, "Advances in combinatorial optimization: Linear programming formulations of the traveling salesman and other hard combinatorial optimization problems," *World Scientific*, 2016.

[17] A. Chaudhuri and K. De, "Fuzzy multi-objective linear programming for traveling salesman problem," *African Journal of Mathematics Computer Science Research*, vol. 4, pp. 64–70, 2011.

[18] M. Mahi, Ö. K. Baykan, and H. Kodaz, "A new hybrid method based on particle swarm optimization, ant colony optimization and 3-Opt algorithms for traveling salesman problem," *Applied Soft Computing*, vol. 30, pp. 484–490, 2015.

[19] Y. Wang and P. Y. Yin, "An approximate algorithm for triangle tsp with a four-vertex-three-line inequality," *International Journal of Applied Metaheuristic Computing*, vol. 6, no. 1, pp. 35–46, 2015.

[20] S. Deb, S. Fong, Z. Tian, R. K. Wong, S. Mohammed, and J. Fiaidhi, "Finding approximate solutions of NP-hard optimization and TSP problems using elephant search algorithm," *The Journal of Supercomputing*, vol. 72, no. 10, pp. 1–33, 2016.

[21] G. Zhang, M. Gheorghe, and J. Cheng, "An approximate algorithm combining p systems and ant colony optimization for taraveling salesman problems," *Miguel Angel Martłnez Del Amor*, pp. 321–340, 2010.

[22] S. S. S. Srivastava, S. Kumar, R. C. Garg, and P. Sen, "Generalized traveling salesman problem through n sets of nodes," *Cors Journal*, 1969.

[23] M. F. Tasgetiren, P. N. Suganthan, and Q. K. Pan, "An ensemble of discrete differential evolution algorithms for solving the generalized traveling salesman problem," *Applied Mathematics and Computation*, vol. 215, no. 9, pp. 3356–3368, 2010.

[24] M. F. Tasgetiren, P. N. Suganthan, and Q. Pan, "A discrete particle swarm optimization algorithm for the generalized traveling salesman problem," in *Proceedings of the 9th Annual Genetic and Evolutionary Computation Conference (GECCO '07)*, pp. 158–167, New York, NY, USA, July 2007.

[25] M. N. Rice and V. J. Tsotras, "Engineering generalized shortest path queries," in *Proceedings of the 2013 29th IEEE International Conference on Data Engineering (ICDE 2013)*, pp. 949–960, April 2013.

[26] N. Ascheuer, M. Jünger, and G. Reinelt, "A branch & cut algorithm for the asymmetric traveling salesman problem with precedence constraints," *Computational Optimization and Applications*, vol. 17, no. 1, pp. 61–84, 2000.

[27] C. Moon, J. Kim, G. Choi, and Y. Seo, "An efficient genetic algorithm for the traveling salesman problem with precedence constraints," *European Journal of Operational Research*, vol. 140, no. 3, pp. 606–617, 2002.

[28] X. Wang and A. C. Regan, "The traveling salesman problem with separation requirements," *European Journal of Operational Research*, vol. 140, no. 5, 2002.

[29] M. Karan and N. Skorin-Kapov, "A branch and bound algorithm for the sequential ordering problem," in *MIPRO, 2011 Proceedings of the 34th International Convention*, pp. 452–457, IEEE, 2011.

[30] X. Cao, L. Chen, G. Cong, J. Guan, N. Phan, and X. Xiao, "KORS: Keyword-aware optimal route search system," in *29th IEEE International Conference on Data Engineering, ICDE 2013*, pp. 1340–1343, Brisbane, Australia, April 2013.

[31] S. Shang, L. Chen, Z.-W. Wei, D.-H. Guo, and J.-R. Wen, "Dynamic shortest path monitoring in spatial networks," *Journal of Computer Science and Technology*, vol. 31, no. 4, pp. 637–648, 2016.

[32] S. Shang, L. Chen, Z. Wei, C. S. Jensen, J.-R. Wen, and P. Kalnis, "Collective travel planning in spatial networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 5, pp. 1132–1146, 2016.

Hindawi
Complexity
Volume 2019, Article ID 4906903, 11 pages
https://doi.org/10.1155/2019/4906903

Research Article

Sign Prediction on Unlabeled Social Networks Using Branch and Bound Optimized Transfer Learning

Weiwei Yuan,[1,2] Jiali Pang,[1] Donghai Guan,[1] Yuan Tian,[3] Abdullah Al-Dhelaan,[4] and Mohammed Al-Dhelaan[4]

[1]*College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China*
[2]*Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing 210093, China*
[3]*School of Computer Engineering, Nanjing Institute of Technology, Nanjing 211816, China*
[4]*Dept. of Computer Science, King Saud University, Riyadh, Saudi Arabia*

Correspondence should be addressed to Donghai Guan; dhguan@nuaa.edu.cn

Received 30 November 2018; Accepted 19 January 2019; Published 14 February 2019

Guest Editor: Jianxin Li

Sign prediction problem aims to predict the signs of links for signed networks. Currently it has been widely used in a variety of applications. Due to the insufficiency of labeled data, transfer learning has been adopted to leverage the auxiliary data to improve the prediction of signs in target domain. Existing works suffer from two limitations. First, they cannot work if there is no target label available. Second, their generalization performance is not guaranteed due to that fact that the solution of their objective functions is not global optimal solution. To solve these problems, we propose a novel sign prediction on unlabeled social networks using branch and bound optimized transfer learning (SP_BBTL) sign prediction model. The main idea of SP_BBTL is to use target feature vectors to reconstruct source domain feature vectors based on relationship projection, which is a complicated optimal problem and is solved by proposed optimization based on branch and bound that can obtain global optimal solution. With this design, the target domain label information is not required for classifier. Finally, the experimental results on the large scale social signed networks validate the superiority of the proposed model.

1. Introduction

Sign prediction predicts signs for links of signed networks, in which signed networks are networks whose edges have signs representing the relationship between nodes. The sign of a link is either positive or negative. A link with a positive sign is also called a positive link, which means the two end nodes of this link trust or like each other. A link with a negative sign is also called a negative link, which means the two end nodes of this link distrust or dislike each other. Compared with unsigned networks which only contain values representing the existence of links, signed networks contain more valuable node relationship information. Because of this rich preserved information, signed networks have been widely used in many applications, such as recommender systems [1, 2] and community detection [3, 4].

Most link prediction methods for signed networks are supervised or semisupervised. It is difficult for them to predict unlabeled target networks without any prior target label information. As for those link prediction methods using transfer learning or ensemble learning technologies, there is nonneglectable knowledge loss in knowledge transferring or domain mapping. The main challenge of the link prediction in signed networks is the data insufficiency problem. And this is the motivation to use an auxiliary labeled network to predict signs for unlabeled target network. Signs are manually labeled by experts, which is time consuming and expensive. This leads to the insufficient number of labeled signs in the real applications. Transfer learning, which is able to transfer knowledge from other domains to assist sign prediction, has therefore been used to address this problem [5, 6]. The domain containing the signs for prediction is called the target

domain, and the domain whose knowledge is transferred to the target domain is called the source domain.

Though transfer learning based sign prediction methods have good performances on labeled signed networks, they are unable to predict signs on unlabeled signed networks. In [7–9], they map the feature vectors both in source domain and target domain into a high dimensional space to get the common knowledge as the transferable knowledge. These mapping approaches lost knowledge in the calculation course of high dimensional space like reproducing kernel Hilbert space. And these methods also lost knowledge in inverse transformation when calculating the reconstruction errors. Existing methods need a number of labeled signs of the target domain to train the sign classifier. Labeled signs of the target domain are used together with the labeled signs of the source domain to map feature vectors of both domains. These feature vectors are mapped to a common feature vector space, and the mapped feature vectors are called the common knowledge for transferring. The common knowledge along with the labeled signs of both domains is then used to train the sign classifier. However, in the real applications, it is sometimes unable to get any labeled signs of the target domain, which makes it impossible for the exiting methods [10] to map both domains for the common knowledge.

In addition, sign prediction performances of existing works need further improvement since the optimization of existing objective functions always lead to local optimal solutions or ill-condition solutions. Transferring knowledge between different domains is a complicated process, so the objective functions of sign prediction are usually nonconvex. Most existing works like [11] use gradient descent algorithms to optimize their nonconvex objective functions to predict signs. However, since the integration length of the existing works is fixed for gradient descent algorithms, it is not always possible for the optimal solution to be selected as the final solution of the objective function. This leads to the local optimal solution of these works' objective functions. So, the sign prediction performances of these works are not stable for use. Some other existing works [12–14] optimize their objective function with least angle regression methods or iterations. However, when getting the analytic solution for the objective function, the error of mapping transformations is usually large with these optimal methods [15–17]. This leads to the great loss of the common knowledge for knowledge transferring.

To solve the problems of existing works, we proposed a novel sign prediction model using branch and bound optimized transfer learning (SP_BBTL). SP_BBTL is different from existing works [18–20] which rely on the target labels to establish relationship between source domain and target domain; SP_BBTL establishes a direct projection from source feature vectors to target feature vectors to obtain the reconstruction errors as the relationship between source domain and target domain. Direct mapping can preserve more original and specific information and knowledge in source domain and it is enough to train the classifier and predict target labels without any prior information of target labels. Besides, SP_BBTL adapts branch and bound optimization to calculate the global optimal solution. Specifically, the proposed model optimizes the objective functions via branch and bound (BB), which can get the global optimal solution by ensuring the bounds of solutions and BB can be applied in many combination optimization problems.

There are three main advantages in SP_BBTL. First, it does not require any sign labels in the target domain because of feature vectors mapping. Secondly, the BB based model can be used to compute the global optimal solution of a nonconvex mixed optimization problem with feature vectors in social networks. Third, the proposed method performs well in the imbalance networks compared with existing works because SP_BBTL gets the global optimal solution in the course of source feature vectors reconstruction that has preserved more complete and original transferable knowledge in source domain.

The rest of this paper is organized as follows. Section 2 gives a brief review of the related works; Section 3 presents the details of the proposed method; Section 4 demonstrates the experimental results; Section 5 concludes this paper and points out the future works.

2. Related Works

In this paper, we propose a novel sign prediction method via transfer learning technology. Thus, the relative works are mainly separated into two parts: sign prediction and transfer learning.

2.1. Sign Prediction. There are mainly three categories for sign prediction approaches. The first type constructs the nonbayesian model based on a set of vertex attributes. The second type derives the joint probability of each sample based on the knowledge of probabilities. The third type leverages linear algebra methods to calculate the similarities between network nodes based on rank-reduced similarity matrices. References [7–9, 21, 22] are supervised which requires a sufficient number of training samples to construct the sign prediction model. All of the existing approaches require some prior knowledge to train classifiers, yet the cost of getting the prior knowledge is expensive in the real applications. Besides, many sign prediction problems face up to class imbalanced problem in reality. Reference [23] is not suitable for the class imbalanced problem, yet [24] utilized adjacency matrix and Laplace matrix to train and test the classifier. But the objective functions in this approach are optimized by iteration computing, which generates considerable error in calculation course.

In addition to model design, another focus of sign prediction is the extraction of useful feature vectors to construct the sign prediction model. There are mainly two types of features: vertex features and edge features. Vertex features consist of neighborhood node based features, path based features, Katz value [25, 26], cluster coefficient scores, etc. Edge features are actually the features of a pair of nodes, which mainly include kernel features conjunction, extended graph formation, and generic SimRank.

2.2. Transfer Learning. In the real social networks, it is very hard or expensive to obtain the label for our target problem

which results in the insufficiency of available data. To solve this problem, transfer learning has been adopted in the sign prediction problem, which tries to utilize the knowledge from source domain to predict the signs in the target domain. Currently transfer learning based sign prediction approaches can be divided into three types: transferring knowledge of instances, transferring knowledge of parameters, and transferring knowledge of feature representations [27]. Reference[28] belongs to instances based methods but they cannot work without target labels. The approaches of transferring knowledge of parameters [29] assume that the model parameters of related learning task can always be shared, which is actually hard for the real networks.

To deal with data insufficiency problem, there are some unsupervised transfer learning approaches [12, 30, 31], which do not require any sign label in the target domain. But they require designing the pivot feature to achieve the good performance for the model. Unfortunately, the design of pivot feature is usually very challengeable. Another related approach SO [32] showed good performance on the regular datasets, yet it has a substantial performance degradation for the class imbalanced datasets.

In general, the analysis of related work shows that traditional sign predictions require a sufficient number of sign labels for training. To alleviate this, transfer learning based approaches have been proposed, yet most of these approaches still need some number of sign labels in the target domain. The existing unsupervised sign prediction approaches based on transfer learning do not need any sign labels in the target domain, but they are usually designed to solve a certain sign prediction problem and hard to use as a universal solution. Therefore, a novel transfer learning based approach for sign prediction is required. In this paper, we propose a novel sign prediction model named sign prediction on unlabeled social networks using branch and bound optimized transfer learning (SP_BBTL). The detailed introduction of SP_BBTL is presented in next section.

3. The Proposed Approach

3.1. Problem Definition. A *signed network* can be represented as a directed graph $\mathbf{G} = (\mathbf{V},\mathbf{E},\mathbf{Y})$, where $\mathbf{V}$ represents the nodes, $\mathbf{E}$ represents the edges, and $\mathbf{Y}$ is the sign of $\mathbf{E}$. $\exists v_i,\ v_j \epsilon \mathbf{V}$, if there is an edge pointing from v_i to v_j, $e_{ij} = 1$, $e_{ij} \epsilon \mathbf{E}$; if there is no connection between v_i and v_j, $e_{ij} = 0$. If $e_{ij} = 1$, and v_i trusts or likes v_j, $y_{ij} = 1$, $y_{ij} \epsilon \mathbf{Y}$; if $e_{ij} = -1$, and v_i distrusts or dislikes v_j, $y_{ij} = -1$, $y_{ij} \epsilon \mathbf{Y}$. An *adjacency matrix* $\mathbf{A}$ is used to describe the connection of $\mathbf{G}$, in which $a_{ij} \epsilon \mathbf{A}$ is the connection between v_i and v_j, here $a_{ij} = y_{ij}$.

Sign prediction predicts y_{ij} for e_{ij} ($e_{ij} \epsilon \mathbf{E}$) of $\mathbf{G}$. To predict signs of links, a feature vector $\mathbf{F}$ is extracted from $\mathbf{A}$ to described $\mathbf{E}$. $\mathbf{F}$ is used to train the sign classifier and then predict signs of target links. *Link prediction* is a learning task that predicts whether a link exists in a labeled or unlabeled network. Sign prediction is a learning task that predicts the signs of links, which also is called labels or weights of links. Labels used in this work consist of positive label and negative label. Predicting links of a network is the same as predicting the labels of links.

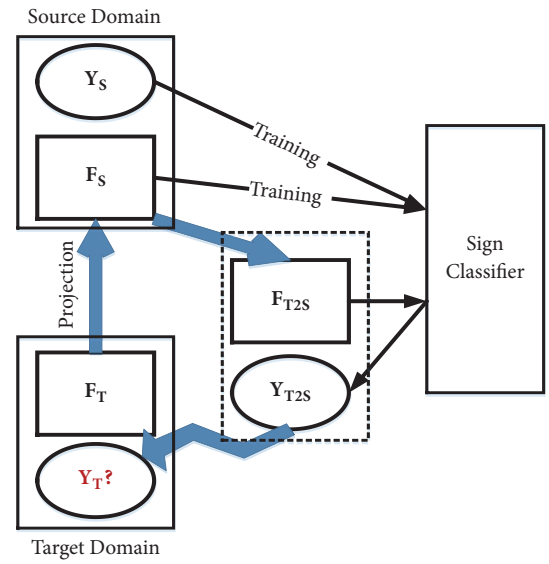


Figure 1: Collaborative representation of the source domain knowledge and the target domain knowledge for sign prediction.

Transfer learning based sign prediction transfers the knowledge of the source domain to the target domain to predict the signs of links for the target domain. Let two signed networks $\mathbf{S} = (\mathbf{V_S}, \mathbf{E_S}, \mathbf{Y_S})$ and $\mathbf{T} = (\mathbf{V_T}, \mathbf{E_T}, \mathbf{Y_T})$ denote the *source domain* and the *target domain* of transfer learning based sign prediction, in which $|\mathbf{V_S}| = n_s$, $|\mathbf{V_T}| = n_T$, $|\mathbf{E_S}| = m_s$, $|\mathbf{E_T}| = m_T$, and $n_S, n_T, m_s, m_T \epsilon N^+$. For $e_{ij} \epsilon \mathbf{E_T}$, SP_BBTL predicts y_{ij} ($y_{ij} \epsilon \mathbf{Y_T}$) for $\mathbf{T}$ with the collaborative representation of $\mathbf{F_S}$, $\mathbf{F_T}$, and $\mathbf{Y_S}$.

3.2. The Proposed SP_BBTL Model. The main idea of the proposed SP_BBTL model is presented in Figure 1. SP_BBTL first constructs a sign classifier based on the knowledge of $\mathbf{S}$, i.e., $\mathbf{F_S}$ and $\mathbf{Y_S}$. SP_BBTL discovers the relationship between $\mathbf{F_T}$ and $\mathbf{F_S}$. This relationship projects $\mathbf{F_T}$ to $\mathbf{F_S}$ and generates a new representation $\mathbf{F_{T2S}}$, which is used to establish the relationship between $\mathbf{G_S}$ and $\mathbf{G_T}$. $\mathbf{F_{T2S}}$ is then used as the input of the trained sign classifier to get the output: predicted signs $\mathbf{Y_{T2S}}$.

The key step of SP_BBTL is to achieve domain adaption from $\mathbf{G_T}$ to $\mathbf{G_S}$ and to establish a mapping from $\mathbf{F_T}$ to $\mathbf{F_S}$:

$$\mathbf{F_{T2S}} = \mathbf{H}\left(\mathbf{F_S}, \mathbf{F_T}\right) \tag{1}$$

where $\mathbf{F_{T2S}}$ is the projection of $\mathbf{F_T}$ into $\mathbf{F_S}$, and $\mathbf{H}$ is the mapping function. The mapping in (1) should maximize the similarity while minimize the difference between $\mathbf{G_S}$ and $\mathbf{G_T}$.

The detailed architecture of the proposed SP_BBTL model is shown in Figure 2, in which the grey rectangle represents the functional module and the white rectangle represents the data. The inputs of SP_BBTL are the adjacency matrices of $\mathbf{T}$ and $\mathbf{S}$. The output is the predicted signs $\mathbf{Y}$ of $\mathbf{T}$. Feature vectors extraction module extracts $\mathbf{F_T}$ and $\mathbf{F_S}$ from the input matrices. A branch bound option based domain reconstruction module is proposed to establish the mapping from $\mathbf{F_T}$ to $\mathbf{F_S}$. The reconstructed feature vectors $\mathbf{F_{T2S}}$ will

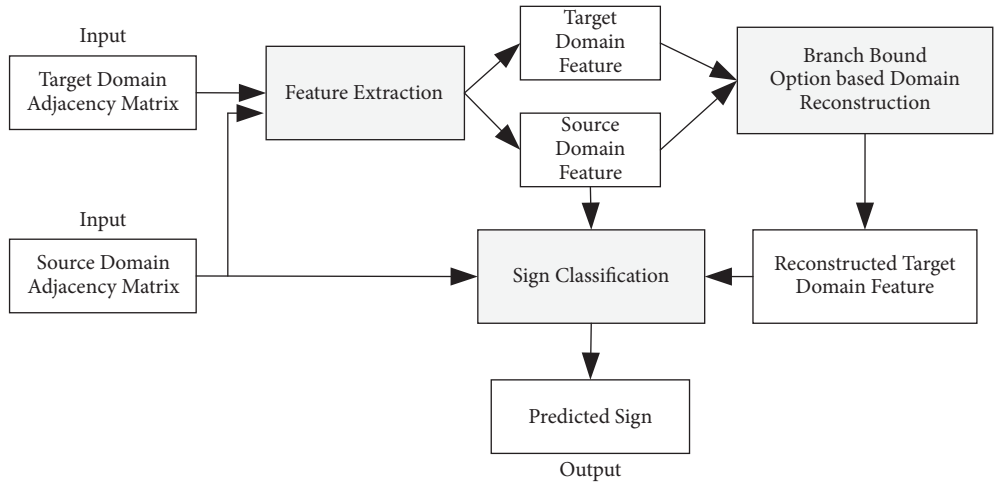


Figure 2: The architecture of the proposed SP_BBTL model.

be used collaboratively with $\mathbf{S}$ to predict $\mathbf{Y_T}$ by the sign classification module. The technical details of SP_BBTL are given in Figure 2.

3.2.1. Feature Vectors Extraction. As shown in Figure 2, given the adjacency matrices of source domain and target domain, feature vectors $\mathbf{F_S}$ and $\mathbf{F_T}$ are firstly extracted to describe links of $\mathbf{S}$ and $\mathbf{T}$ for link prediction. In this work, five features are extracted for each feature vectors. These features include link positive outdegree, link negative outdegree, link positive indegree, link negative indegree, and link embeddedness [33].

Link positive outdegree $d_{out+}(v_i)$ denotes the number of positive edges pointing from v_i to other nodes. $d_{out+}(v_i)$ reflects the likelihood that v_i gives positive sign to a connected link. The higher value $d_{out+}(v_i)$ has, the more probably $y_{ij} = 1$. Link negative outdegree $d_{out-}(v_i)$ is the number of negative edges pointing from v_i to other nodes. $d_{out-}(v_i)$ reflects the likelihood that v_i gives negative sign to a connected link. The higher value $d_{out-}(v_i)$ has, the more probably $y_{ij} = -1$.

Link positive indegree $d_{in+}(v_j)$ is the number of positive edges pointing to v_j. $d_{in+}(v_j)$ reflects the likelihood that v_j gets positive sign from a connected link. The higher value $d_{in+}(v_j)$ has, the more probably $y_{ij} = 1$. Link negative indegree $d_{in-}(v_j)$ is the number of negative edges pointing to v_j. $d_{in-}(v_j)$ reflects the likelihood that v_j gets negative sign from a connected link. The higher value $d_{in-}(v_j)$ has, the more probably $y_{ij} = -1$.

Link embeddedness $\mathrm{em}(e_{ij})$ is the number of common neighbors of v_i and v_j. The link embeddedness of each edge (or link) contains the essential characteristic relationship among its neighbor nodes, which reflects the global structural feature of a substructural network in the whole network. $\mathrm{em}(e_{ij})$ also represents the structural balance of e_{ij}: according to the structural balance theory, the higher value $\mathrm{em}(e_{ij})$ has, the more probably a positive relationship between v_i and v_j exists and the more probably $y_{ij} = 1$.

3.2.2. Branch and Bound Optimized Domain Reconstruction. Domain reconstruction is the key part of SP_BBTL model. It will build up the latent relationship between source feature vectors and target feature vectors collaboratively. Reconstructing domain from $\mathbf{F_T}$ to $\mathbf{F_S}$ can be represented as

$$\mathbf{F_T X} = \mathbf{F_S} \tag{2}$$

where $\mathbf{X}$ is the solution of (2). In essence, $\mathbf{X}$ is the bridge to enable the collaborative use of knowledge in $\mathbf{S}$ and $\mathbf{T}$. $\mathbf{X}$ is denoted by $\mathbf{F_{T2S}}$ in Figure 1, and (2) can be also written as

$$\mathbf{F_T F_{T2S}} = \mathbf{F_S} \tag{3}$$

A branch and bound based method is proposed for solving (3) with the minimum globalized error to get a global optimal solution. With minimum error, (3) can be rewritten as

$$\boldsymbol{X}^* = \operatorname{argmin} \left\| \mathbf{F_S} - \mathbf{F_T X} \right\|_1, \tag{4}$$

where $\boldsymbol{X}^*$ is the optimum solution of (3). 1-norm sums the matrix along the column to select the maximum numerical value. If the sum of each column of $\mathbf{X}$ is minimized, the reconstruction error is minimized. This ensures the divergence of the transfer learning task to be minimized.

To minimize the error of (4), motivated by the idea of sparse coding, the constrained condition $\mathscr{g}$ is set to be

$$\mathscr{g}: \left\| \boldsymbol{X}_{(i)} \right\|_{\mathbf{0}} \le \alpha, \quad i = 1, 2, \ldots, n_T \tag{5}$$

where α is the number of nonzero elements in each column of $\mathbf{X}$. $\mathscr{g}$ can control the sparsity of $\mathbf{F_S}$ for the reconstruction. α ensures the nonzero elements corresponding to the selected samples are neighbors of $\mathbf{X}$.

Calculating (2) is solving a problem of mixed optimization. However, existing methods can only calculate the local optimal solution, which cannot get the global optimal solution of (2). Therefore, the branch and bound method is

Input: $\mathbf{F_T}$ and $\mathbf{F_S}$
Output: Global optimal value of $\mathbf{X}$, reconstruction error e
Parameters: Constraint parameter α,
Cut-off error ε,
ϕ_{lb}: lower bound function;
ϕ_{ub}: upper bound function;
Q_{Init}: a n-dimensional vectors set.
1. Initialize $\mathbf{X}$ in (4)
2. Calculate the lower bound L and upper bound U on $\mathbf{X}$:
3. while $U - L \geq \varepsilon$ do
4. $L = \phi_{lb}(Q_{Init}), U = \phi_{ub}(Q_{Init})$
5. end while
6. Split Q_{Init} into 2 number sets Q_1 and Q_2, where $Q_{Init} = Q_1 \cup Q_2$
7. Compute $\phi_{lb}(Q_i)$ and $\phi_{ub}(Q_i), i = 1, 2,$
8. Update lower and upper bounds of $\mathbf{X}$.
9. while $U - L \geq \varepsilon$ do
10. $L = \min\{\phi_{lb}(Q_1), \phi_{lb}(Q_2)\}$
11. $U = \min\{\phi_{ub}(Q_1), \phi_{ub}(Q_2)\}$
12. end while
13. Refine partition via splitting Q_1 or Q_2.
14. if $U - L \geq \varepsilon$ or $\|\boldsymbol{X}_{(i)}\|_0 \leq \alpha$ then
15. repeat step 6 and 7
16. else
17. return $\mathbf{X}$, e
18. end if
19. Update the value of $e = \|\mathbf{F_S} - \mathbf{F_T X}\|$

Algorithm 1: Branch and Bound optimized Domain Reconstruction.

proposed to achieve the optimal solution calculation. Branch and bound is a generalized search algorithm which includes searching and iterating. Specifically speaking, it compares the size relationship between the given error ε and the difference about upper bound and lower bound of the feature vectors, and then it adjusts the upper bound and lower bound according to the size relationship. This method controls the complexity of solution vectors in different network scales via parameter α. It can calculate out the global optimal solution for mixed optimization problem. [34] The details are shown in Algorithm 1.

3.2.3. Sign Classification. With the extracted global optimal solution $\mathbf{X}$, which is $\mathbf{F_{T2S}}$ in (3), $\mathbf{Y_T}$ is predicted as

$$\mathbf{Y_T} \approx \mathbf{Y_{T2S}} = \mathbf{C}\left(\mathbf{F_{T2S}}, \mathbf{F_S}, \mathbf{Y_S}\right), \tag{6}$$

where $\mathbf{C}$ is a sign classifier that is trained by $\mathbf{F_S}$ and $\mathbf{Y_S}$ and $\mathbf{F_{T2S}}$. $\mathbf{Y_{T2S}}$ is the output of $\mathbf{C}$ to get the predicted sign $\mathbf{Y_{T2S}}$.

4. Experimental Results

Four datasets extracted from the real-world applications are used in the experiments to verify the performances of the proposed method. These datasets (http://snap.stanford.edu/data/index.html) are Bitcoinotc [35] (denoted as OTC), Bitcoinalpha (denoted as ALP) [35], Epinions (denoted as EPI)[36], and Slashdot (denoted as SLA)[37]. The values of data label for EPI and SLA belong to {-1,1}. The values of data labels for OTC and ALP are mapped into {-1,1} by setting the label to be -1 if the original label is less than 0 and setting the label to be 1 if the original label is greater than 0. The details of the experimental datasets are given in Table 1. Accuracy and F1-score [38] are used to measure the sign prediction performances of the proposed method.

Two baseline methods are used in this paper to compare with SP_BBTL. The first method is Source-Only (SO) model. It predicts signs of links with data merely from source domain [32]. The second method is Nonnegative Matrix Trifactorization (NMTF) [14]. NMTF predicts signs of links by matrix trifactorization using source domain labels and target domain feature vectors. By factorizing the adjacency matrix of the source domain and the target domain, NMTF gets the latent feature vectors of each domain, which are used together with the explicit feature vectors of each domain to transfer knowledge from the source domain to the target domain.

Sign prediction performances of SP_BBTL are firstly measured with various network sizes. In the experiments, the number of samples in the target domain is fixed to 3000, while the number of source domain samples varies from 3500 to 9500. The proportion of positive link to negative link is set to be 7:3 (± 5%). The experimental results are given in Figure 3, in which AAA-BBB means AAA is the source domain of sign prediction and BBB is the target domain of sign prediction. For example, OTC-ALP means predicting signs of ALP by using OTC as the source domain.

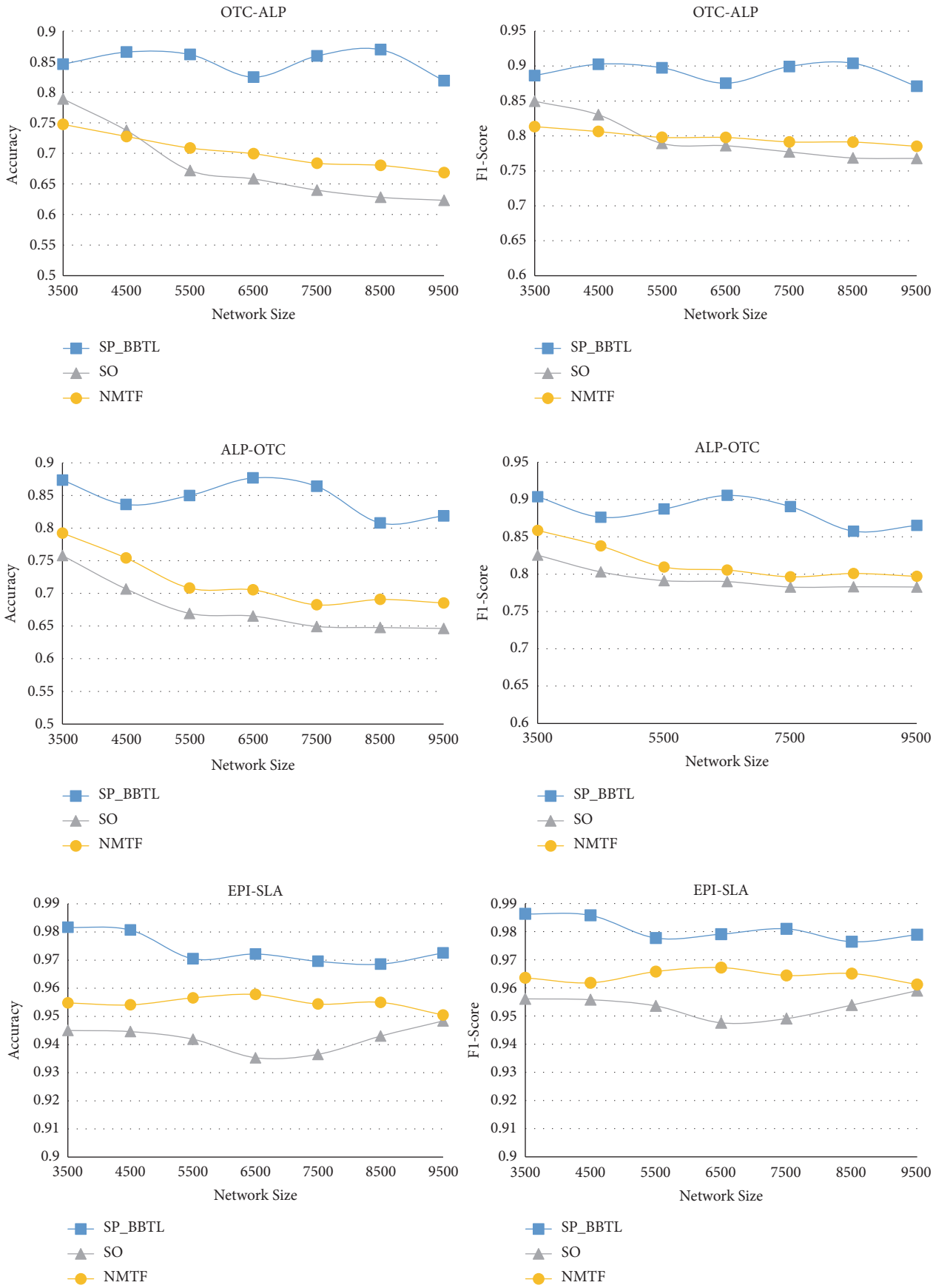


Figure 3: Continued.

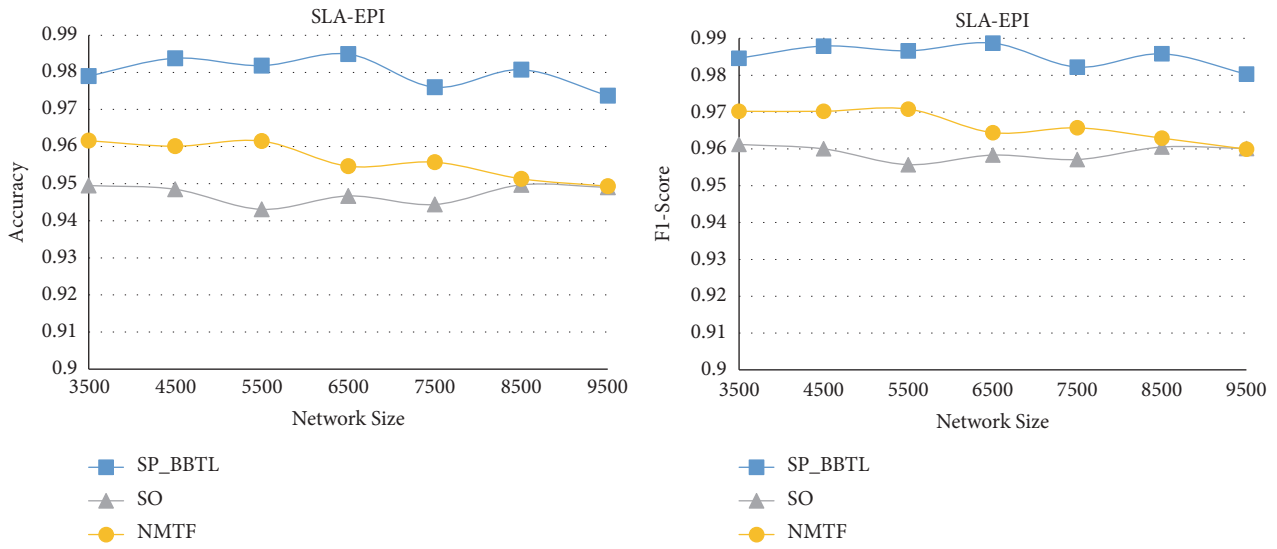


Figure 3: Sign prediction performances with the various network sizes.

Table 1: Detail information of the experimental datasets.

	OTC	ALP	EPI	SLA
Number of nodes	5881	3783	131828	82140
Number of links	35592	24186	841372	549202
Average degree	12.1041	12.7867	13.3723	12.7647
Number of negative links	11981	8890	123705	124130
Negative link ratio	33.66%	36.76%	14.7%	22.6%

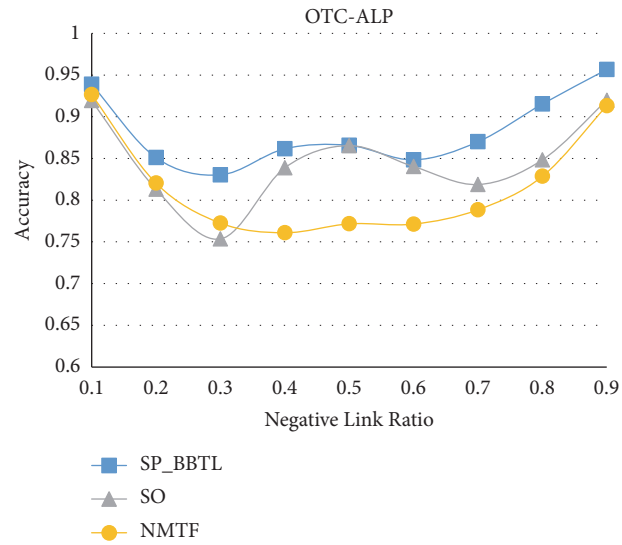
As shown in Figure 3, when transferring knowledge from each of the source domain-target domain groups, the tendency of the performance about network size is slightly decreasing because the accuracy is negative related to network size. The proposed SP_BBTL performs better than the baselines because SP_BBTL really can transfer more useful knowledge from source domain to target domain. In addition, the more efficient optimization also contributes to the superior sign prediction performance of SP_BBTL. Compared with our proposed model, the two baseline methods failed to decrease the transfer loss. This means the solution of their objective function is not globally optimal, which leads to their limited link prediction performances.

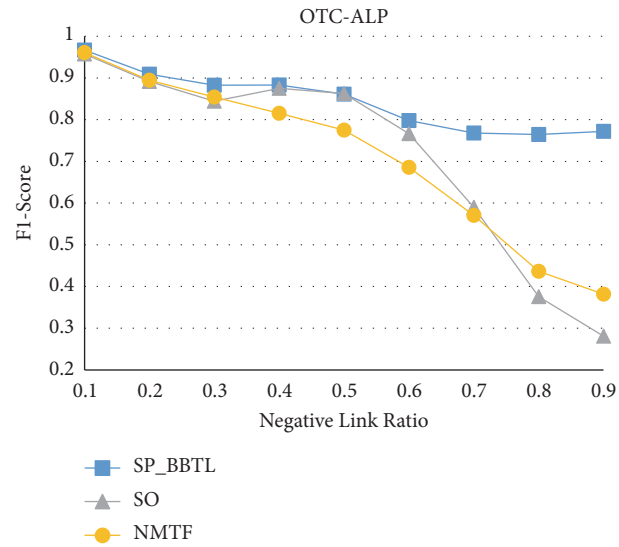
Sign prediction performances of SP_BBTL are then measured with the various negative link ratios. In the experiments, the number of samples in the target domain is set to be 3000, while the number of samples in the source domain is set to be 4500 (OTC-ALP and ALP-OTC) and 6500 (EPI-SLA and SLA-EPI) respectively. The ratio of negative links varies from 10% to 90%. The experimental results are given in Figure 4. It is shown that the accuracy and F1-score of SP_BBTL are superior to baselines with different negative link ratios. The accuracy of SP_BBTL and the baseline method tends to be micro-W-like distribution on OTC-ALP and ALP-OTC dataset, while the accuracy of SP_BBTL and the baseline method tends to be micro-V-like distribution on EPI-SLA and SLA-EPI datasets. F1-score of SP_BBTL and baseline methods decreases with the increasing of negative link ratios. In addition, SP_BBTL is insensitive to the decreasing of negative link ratios, while the baseline methods decrease significantly with the increasing of negative link ratios, especially on OTC-ALP and ALP-OTC datasets.

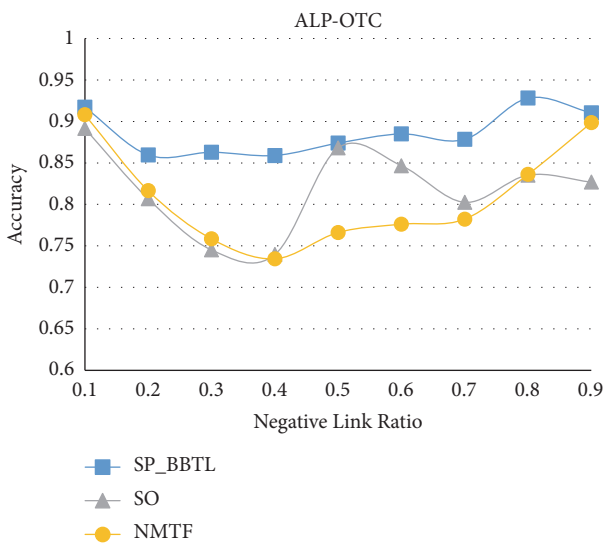
The influence of the constraint parameter α on sign prediction performances of SP_BBTL is further analyzed. In the experiments held for OTC-ALP and ALP-OTC, the scale of the source domain and the scale of the target domain are 4500 and 3000, respectively. In the experiments held for EPI-SLA and SLA-EPI, the scale of the source domain and the scale of the target domain are 6500 and 3000, respectively. The value of the constraint parameter α varies from 0.1 to 2.5, and the negative link ratio is set to be 0.7 in both source domain and target domain. The experimental results are given in Figure 5. Based on the experimental results, the performance of SP_BBTL is relatively stable when α is larger than 0.5. When α is close to 0, that means the zero solution for (3) which is meaningless in sign prediction. So, α is suggested to be a value around 0.5 and this contributes to the best prediction performances of SP_BBTL.

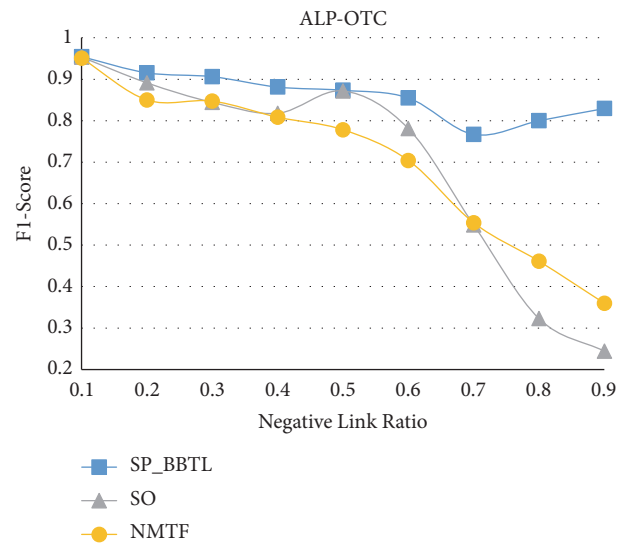
5. Conclusion and Future Work

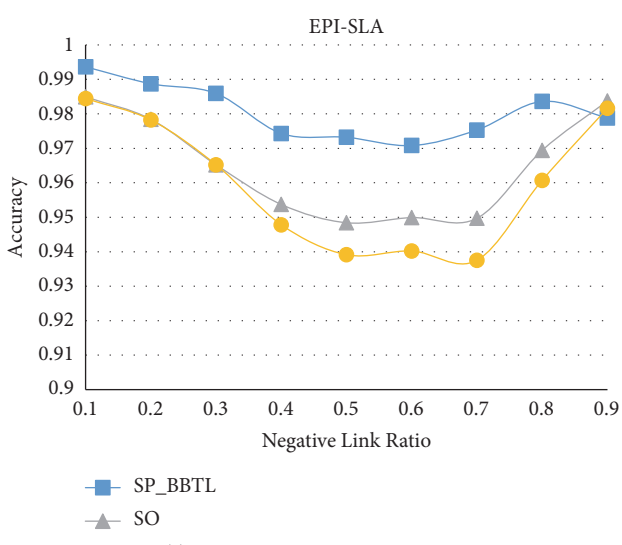
In this paper, a novel method named sign prediction on unlabeled social networks using branch and bound optimized transfer learning (SP_BBTL) is proposed to solve a sign prediction problem via feature vectors projections. In SP_BBTL, labeled source feature vectors are mapped into











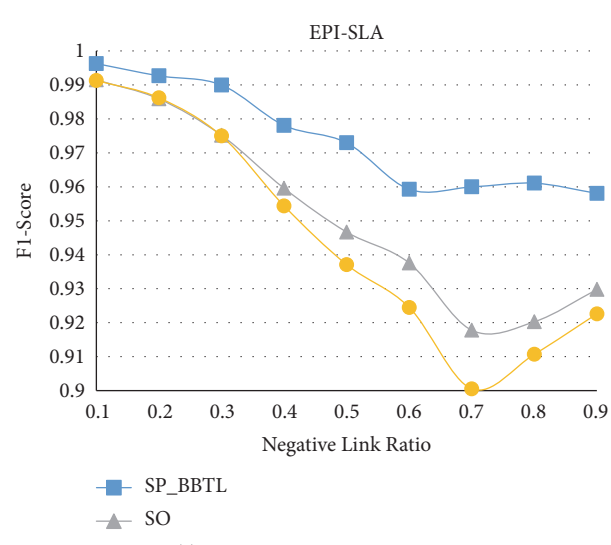


Figure 4: Continued.

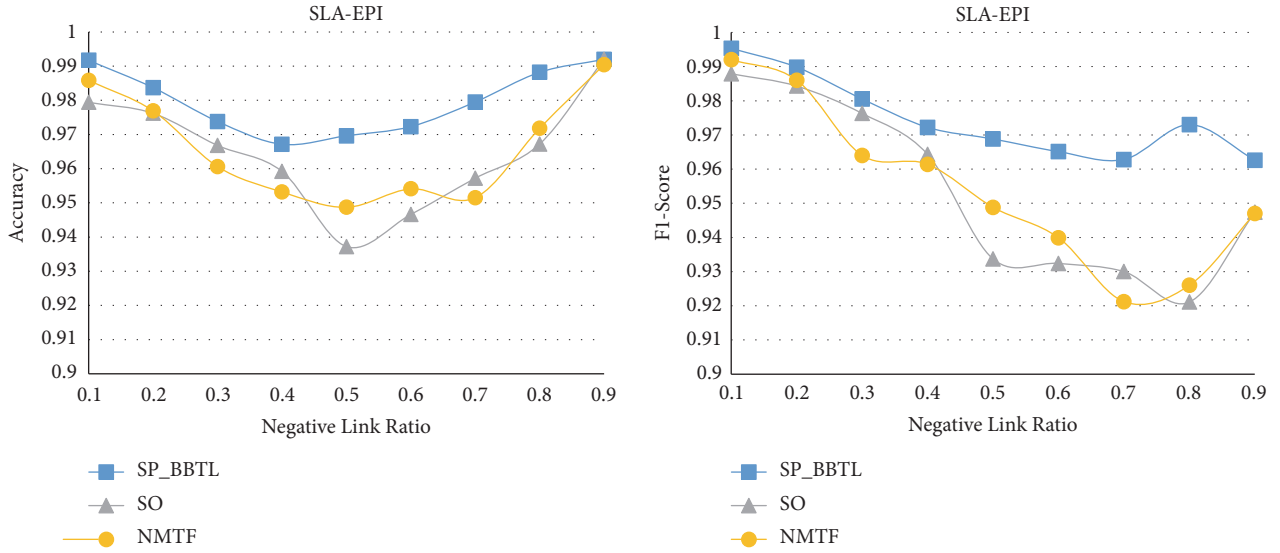


Figure 4: Sign prediction performances with the various negative link ratio.

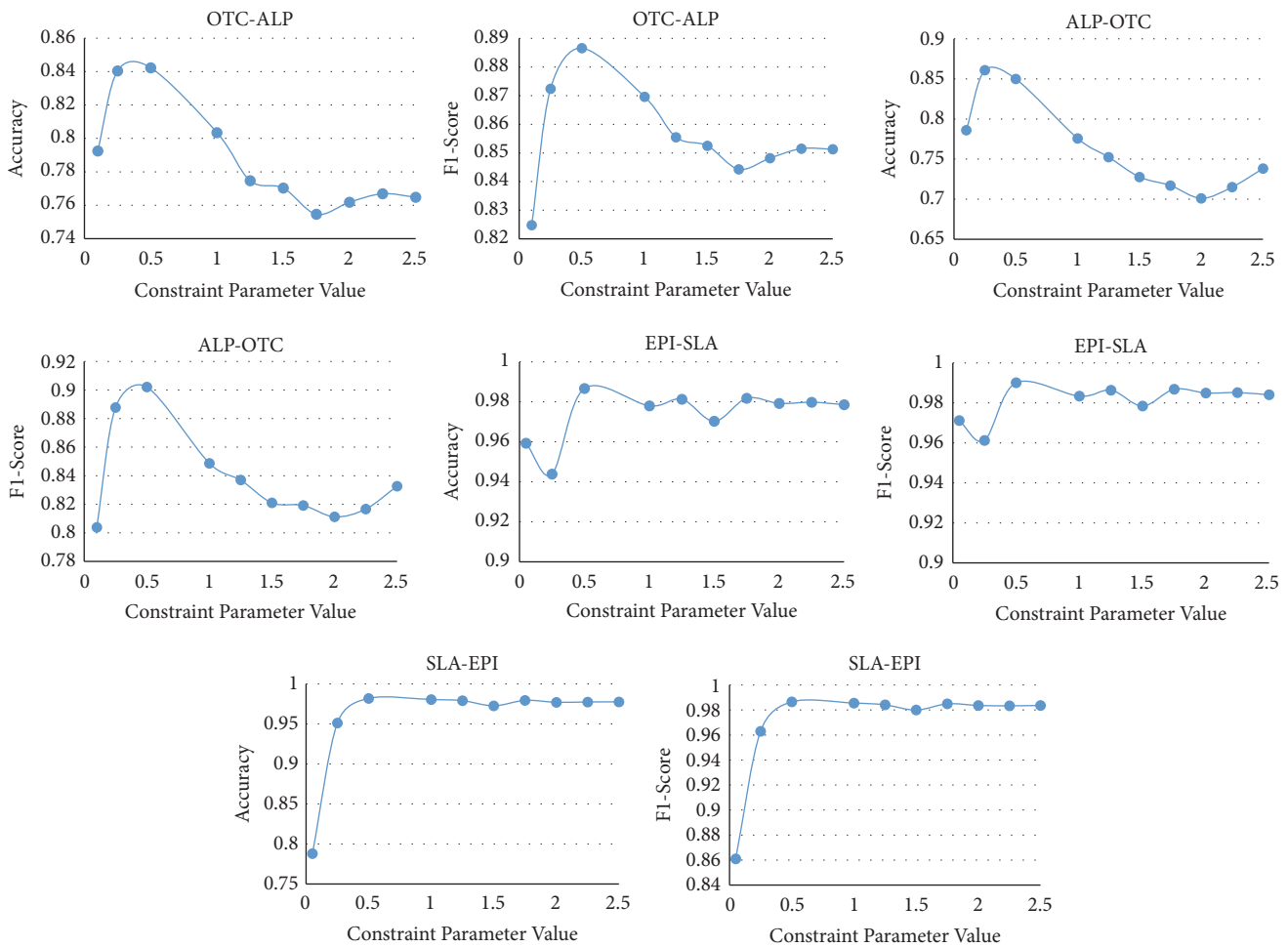


Figure 5: The influence of the constraint parameter on sign prediction performances.

unlabeled target feature vectors and then the relationship between two domains can be established so that the classifier can be trained without any target label. In addition, the proposed optimization based on branch and bound (BB) performs efficient on social networks because the branch and bound optimization method adapted in the proposed model can ensure the global optimal solution of the objective function. Branch and bound can get global optimal solution by highly efficient searching and iteration. It can maximize the transferable knowledge of the source domain, while minimize the transfer loss. Experimental evaluation validates the superior effectiveness and stability of SP_BBTL in real social networks. At last we give the suggested value for parameter α in proposed model.

In the future, we will try to improve the proposed method from several aspects. Firstly, we will try to develop a generalized algorithm, which could not only minimize the influence of negative transfer, but also discover transferable knowledge with different categories of source domains, such as the text data and the image data. Secondly, we will improve the model to minimize the number of the source domain instances used for knowledge transfer, only with little cost in link prediction performances. Lastly, we will extend our model from solving binary sign prediction problem to multilabel sign prediction problem.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This research was supported by Nature Science Foundation of China (Grant No. 61672284), Natural Science Foundation of Jiangsu Province (Grant No. BK20171418), China Postdoctoral Science Foundation (Grant No. 2016M591841), Jiangsu Planned Projects for Postdoctoral Research Funds (No. 1601225C). The authors extend their appreciation to the Deanship of Scientific Research at King Saud University for funding this work through research group no. RGP-VPP-264.

References

[1] E. Gündoğan and B. Kaya, "A recommendation method based on link prediction in drug-disease bipartite network," in *Proceedings of the 2nd International Conference on Advanced Information and Communication Technologies, AICT 2017*, pp. 125–128, Ukraine, July 2017.

[2] P. Moradi and S. Ahmadian, "A reliability-based recommendation method to improve trust-aware recommender systems," *Expert Systems with Applications*, vol. 42, no. 21, pp. 7386–7398, 2015.

[3] V. Lyzinski, M. Tang, A. Athreya, Y. Park, and C. E. Priebe, "Community detection and classification in hierarchical stochastic blockmodels," *IEEE Transactions on Network Science and Engineering*, vol. 4, no. 1, pp. 13–26, 2017.

[4] L. Yang, X. Cao, D. Jin, X. Wang, and D. Meng, "A unified semi-supervised community detection framework using latent space graph regularization," *IEEE Transactions on Cybernetics*, vol. 45, no. 11, pp. 2585–2598, 2015.

[5] C. Yunfang, W. Tongli, and Z. Wei, "Social link prediction based on the users' information transfer," in *Asia-Pacific Web Conference*, pp. 64–76, 2016.

[6] Y. Chen, T. Wang, and W. Zhang, "Link prediction analysis of internet public opinion transfer from the individual perspective," *New Technology of Library Information Service*, 2016.

[7] A. Daniely, R. Frostig, and Y. Singer, "Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity," *Advances In Neural Information Processing Systems*, pp. 2261–2269, 2016.

[8] Y. Zhang, S. Wang, P. Phillips, J. Yang, and T.-F. Yuan, "Three-dimensional eigenbrain for the detection of subjects and brain regions related with alzheimer's disease," *Journal of Alzheimer's Disease*, vol. 50, no. 4, pp. 1163–1179, 2016.

[9] R. Marcotte, A. Sayad, K. R. Brown et al., "Functional genomic landscape of human breast cancer drivers, vulnerabilities, and resistance," *Cell*, vol. 164, no. 1-2, pp. 293–309, 2016.

[10] Z. Shi, P. Siva, and T. Xiang, "Transfer learning by ranking for weakly supervised object annotation," 2017, https://arxiv.org/abs/1705.00873.

[11] O. Sener and V. Koltun, "Multi-task learning as multi-objective optimization," *in in Neural Information Processing Systems*, pp. 524–535, 2018.

[12] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa, "Visual Domain Adaptation: A survey of recent advances," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 53–69, 2015.

[13] M. Kan, J. Wu, S. Shan, and X. Chen, "Domain adaptation for face recognition: Targetize source domain bridged by common subspace," *International Journal of Computer Vision*, vol. 109, no. 1-2, pp. 94–109, 2014.

[14] J. Ye, H. Cheng, Z. Zhu, and M. Chen, "Predicting positive and negative links in signed social networks by transfer learning," in *Proceedings of the 22nd International Conference on World Wide Web*, pp. 1477–1488, ACM, May 2013.

[15] D. Wollmann and M. T. A. Steiner, "The strategic decision-making as a complex adaptive system: a conceptual scientific Model," *Complexity*, vol. 2017, Article ID 7954289, 13 pages, 2017.

[16] P. Liu and F. Teng, "Multiple criteria decision making method based on normal interval-valued intuitionistic fuzzy generalized aggregation operator," *Complexity*, vol. 21, no. 5, pp. 277–290, 2016.

[17] O. Abedinia, N. Amjady, and A. Ghasemi, "A new metaheuristic algorithm based on shark smell optimization," *Complexity*, vol. 21, no. 5, pp. 97–116, 2016.

[18] F. Zhuang, X. Cheng, P. Luo, S. J. Pan, and Q. He, "Supervised representation learning with double encoding-layer autoencoder for transfer learning," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 9, no. 2, p. 16, 2018.

[19] J. Tang, T. Lou, J. Kleinberg, and S. Wu, "Transfer learning to infer social ties across heterogeneous networks," *ACM Transactions on Information Systems (TOIS)*, vol. 34, no. 2, p. 7, 2016.

[20] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," 2016, https://arxiv.org/abs/1605.06636.

[21] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "uriosity-driven exploration by self-supervised prediction," in *International Conference on Machine Learning (ICML)*, pp. 488-489, 2017.

[22] Y. Kuznietsov, J. Stückler, and B. Leibe, "Semi-supervised deep learning for monocular depth map prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6647–6655, 2017.

[23] J. Kunegis and A. Lommatzsch, "Learning spectral graph transformations for link prediction," in *Proceedings of the 26th International Conference On Machine Learning, ICML 2009*, pp. 561–568, Canada, June 2009.

[24] R. Pech, D. Hao, L. Pan, H. Cheng, and T. Zhou, "Link prediction via matrix completion," *EPL (Europhysics Letters)*, vol. 117, no. 3, p. 38002, 2017.

[25] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.

[26] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowledge-Based Systems*, vol. 151, pp. 78–94, 2018.

[27] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[28] R. Wang, M. Utiyama, L. Liu, K. Chen, and E. Sumita, "Instance weighting for neural machine translation domain adaptation," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1482–1488, Copenhagen, Denmark, September 2017.

[29] S. Kumar, X. Gao, and I. Welch, "Learning under data shift for domain adaptation: a model-based co-clustering transfer learning solution," in *Pacific Rim Knowledge Acquisition Workshop*, pp. 43–54, 2016.

[30] P. Peng, T. Xiang, Y. Wang et al., "Unsupervised cross-dataset transfer learning for person re-identification," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 1306–1315, USA, July 2016.

[31] B. Tan, Y. Song, E. Zhong, and Q. Yang, "Transitive transfer learning," in *Proceedings of the 21st ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2015*, pp. 1155–1164, Australia, August 2015.

[32] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," 2014, https://arxiv.org/abs/1409.7495.

[33] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Predicting positive and negative links in online social networks," in *Proceedings of the 19th International World Wide Web Conference (WWW '10)*, pp. 641–650, April 2010.

[34] D. R. Morrison, S. H. Jacobson, J. Sauppe, and E. C. Sewell, "Branch-and-bound algorithms: a survey of recent advances in searching, branching, and pruning," *Discrete Optimization*, vol. 19, pp. 79–102, 2016.

[35] S. Kumar, F. Spezzano, V. S. Subrahmanian, and C. Faloutsos, "Edge weight prediction in weighted signed networks," in *Proceedings of the 16th IEEE International Conference on Data Mining, ICDM 2016*, pp. 221–230, Spain, December 2016.

[36] M. Richardson, R. Agrawal, and P. Domingos, "Trust management for the semantic web," in *International Semantic Web Conference*, pp. 351–368, 2003.

[37] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, "Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters," *Internet Mathematics*, vol. 6, no. 1, pp. 29–123, 2009.

[38] W. Yuan, C. Li, G. Han, D. Guan, L. Zhou, and K. He, "Negative sign prediction for signed social networks," *Future Generation Computer Systems*, 2017.

Hindawi
Complexity
Volume 2019, Article ID 8503962, 8 pages
https://doi.org/10.1155/2019/8503962

Research Article

Discovering Travel Community for POI Recommendation on Location-Based Social Networks

Lei Tang, Dandan Cai, Zongtao Duan, Junchi Ma, Meng Han, and Hanbo Wang

School of information engineering, Chang'an University, Xi'an, Shanxi, 710064, China

Correspondence should be addressed to Lei Tang; tanglei24@gmail.com

Received 27 November 2018; Accepted 3 January 2019; Published 12 February 2019

Guest Editor: Jianxin Li

Point-of-interest (POI) recommendations are a popular form of personalized service in which users share their POI location and related content with their contacts in location-based social networks (LBSNs). The similarity and relatedness between users of the same POI type are frequently used for trajectory retrieval, but most of the existing works rely on the explicit characteristics from all users' check-in records without considering individual activities. We propose a POI recommendation method that attempts to optimally recommend POI types to serve multiple users. The proposed method aims to predict destination POIs of a user and search for similar users of the same regions of interest, thus optimizing the user acceptance rate for each recommendation. The proposed method also employs the variable-order Markov model to determine the distribution of a user's POIs based on his or her travel histories in LBSNs. To further enhance the user's experience, we also apply linear discriminant analysis to cluster the topics related to *"Travel"* and connect to users with social links or similar interests. The probability of POIs based on users' historical trip data and interests in the same topics can be calculated. The system then provides a list of the recommended destination POIs ranked by their probabilities. We demonstrate that our work outperforms collaborative-filtering-based and other methods using two real-world datasets from New York City. Experimental results show that the proposed method is better than other models in terms of both accuracy and recall. The proposed POI recommendation algorithms can be deployed in certain online transportation systems and can serve over 100,000 users.

1. Introduction

The check-in behaviors in location-based social networks (LBSNs) have become a new lifestyle component for millions of users who share their point-of-interest (POI) locations with their contacts in such LBSNs and provide geo-tagged user posts, photos, and micropayments [1]. The functionality of LBSNs has become increasingly sophisticated in recent years and now includes numerous user-centric services. Among these, personalized POI recommendations [2, 3], such as cinemas, restaurants, and tourist attractions, that is predicted to be of personal interest to users, have become an increasingly important service that can greatly enhance the travel experience of users [4, 5]. As a result, POI recommendations that can greatly enhance the travel experience of users have received increasing attention from both industry and academia [3]. However, previous studies usually fit a POI recommendation model based on all the collected check-in data. It does not fully capture user behavior in different scenarios due to the heterogeneities of interuser and intrauser differences [6]. Here, we investigate the tendency of users to travel under different patterns (both spatial and semantic), and also their tendency to select the POIs based on their interests and social links. In addition, existing POI recommendations have generally sought to discover unknown POI types for a user from his or her contacts [7]. However, these recommendations can involve very sparse data and [8], moreover, they fail to make use of the semantics of POI data for LBSN users with similar interests and travel habits as those of the target user, but which are not among the user's contacts.

Based on the above considerations, in this paper we propose a personal POI recommendation method based on destination prediction. First, the check-in behaviors of individuals are analyzed from the distribution of a user's POIs based on their travel histories in LBSNs. A variable-order Markov model is employed to predict the intended POI types [9], with consideration of the semantics in the spatial layout, which serves as the key constraint of the recommendation

process. This overcomes the weaknesses associated with existing POI recommendations, by not only contacting the users with POIs visited, but also identifying the influence of the current users' locations on their future movements and taking the types of the next destination into account as the POI types to be recommended. Second, groups of users [10] with similar preferences based on users' historical trip data and interests in the same topics are obtained [11]. The probability of each POI intended by a user using the suggestions from the communities can thus be calculated. Finally, a list of the recommended destination POIs ranked by probability is provided.

Discovering the travel communities alleviates the problem of data sparsity [12] while simultaneously alleviating the weaknesses of existing methods based on the check-in data collected from the contacts of users, which ignores the fact that many users like interacting with people with different social links [13]. In addition, social interaction has been used in conjunction with movement-related information to improve recommendations by detecting the communities, which enhances the users' experiences. An analysis employing real-world datasets demonstrates that the use of activity pattern and social interaction for real-life communities facilitates a significant increase of the number of candidate POIs, which contributes to more accurate POI recommendations to individuals relative to ratings-based POI recommendations.

Our key contributions are summarized as follows.

(i) We generalize POI recommendations by detecting communities from social interactions and semantics in the spatial movements.

(ii) We solve the personalized POI recommendation by taking the types of each user's historical POIs as the candidates. The prediction model is trained to calculate the probability of users' intended POIs based on departure longitude and latitude and on departure time.

(iii) We evaluate the proposed method against other existing recommendation techniques on two real-world datasets. Experimental results demonstrate that our approach accurately discovers real grouping behaviors, recommends the most interested POIs to the target users in both test cases, and outperforms existing algorithms.

The remainder of this paper is organized as follows. Pertinent research specific to existing POI recommendation techniques employed by LBSNs is presented in Section 2. In Section 3, the POI prediction model, with the integrated travel community, is proposed. The travel-community-based recommendation algorithm is derived in Section 4. In Section 5, numerical results are provided to demonstrate the advantages of the proposed method over two other algorithms. Concluding remarks are given in Section 6.

2. Related Work

The datasets for POI recommendations usually include global-positioning-system- (GPS-) based trajectory and the check-in data of LBSNs [14]. While numerous studies [15–17] have developed POI recommendations based on GPS trajectory data, these approaches first mined the sequence of semantic POIs visited, which is represented by the check-in data. Additionally, check-in data provide additional information markers (e.g., social interactions, POI types, or semantics in the spatial layout) that are especially useful in capturing latent relationships among users of the same POI type. Therefore, a POI recommendation based on check-in data is greatly favored by researchers, and numerous studies have been conducted [1, 18–20] .

Increasingly sophisticated POI recommendations have been developed based on check-in data, although each has characteristic weaknesses. For example Berjani and Strufe [21] applied a collaborative filtering (CF) model with check-in data for conducting a POI recommendation. Unfortunately, differences in the number of times a user checks in at the various locations are ignored, leading to the inability to fully discover and rank the users by their interests based on locations. Shi et al. [22] recommended POI locations based on a category-related regular matrix using the historical locations visited by users. However, without considering the current travel activity of users, that method cannot suggest where a user should go next. Ye et al. [23] used the ratings provided by friends in conjunction with the social distance among friends to provide a POI recommendation, without any information regarding the user's travel interests. Ference et al. [24] extended a CF model with user's current locations and social interactions, but it recommends a POI only from the travel distance and searches a similar user considering only their social influence [25]. Therefore, it did not work well for sparse datasets. A Bayes classification was used by Jing-jin to calculate the check-in probability of users for specific locations in the future using the historical check-in spots, under a distance-based constraint [26]. However, this method did not identify the influences from other users on the recommendation, which accordingly reduced the accuracy of the recommendation. Ye et al. [27] constructed the diffusion process on multiple information sources (i.e., people's interests, social influences, and spatial proximity) to improve the accuracy of their proposed recommendation. However, this method can provide a general list of the intended POIs without regard to the locations in which a user is at the present moment.

3. Travel Community Discovery from Predicted Semantic POI

3.1. Semantic POI Prediction. The sequence of semantic POIs visited is especially useful in capturing latent relationships among community members [28]. A POI-related model describes the temporal activity pattern for real-life users that includes all h ($h > 1$) POIs visited by user u over a 1 d period as $Seq\_loc_u = loc_0, \ldots, loc_1, \ldots, loc_h$, where loc_i represents a spot from the trajectory database, $loc_i = (lat, lon, check\_in_{time}, POI_i, POI\_category_j)$, which is defined according to the latitude (lat) and longitude (lon) of the check-in time ($check\_in_{time}$), the name of the POI (POI_i,

$1 \le i \le M$), and the POI types ($POI\_category_j$ $1 \le j \le M$). All m historical trajectories of user u are collected in $Seq_u = (id, Seq\_Loc_u)$, $id = 1, 2, \ldots, m$, where id distinguishes trajectories.

We applied the variable-order Markov model to predict the POI destination [29]. The set of historical POIs of a user is abstracted from Seq_u that is given as HPOI=POI_i. Given POI_i in Seq_Loc_u, if $POI_i \in HPOI$, then the Nth-order context model loc_n^N of POI_i refers to a sequence of length N with loc_n as the next POI; that is, $loc_n^N = loc_{n-(N-1)}, \ldots, loc_{n-1}, loc_n$. By looking for the trajectories with the same length as that of loc_n^N in Seq_u, we can predict the probability distribution of POI_i from the number of trajectories observed based on the prediction by partial matching (PPM) model. We calculate the probability p_i of *users*' next POI destinations based on the context model using

$$p_i = p\left(POI_i \mid loc_n^N\right) = \frac{f\left(POI_i \mid loc_n^N\right)}{S\left(loc_n^N\right)}, \tag{1}$$

Here, $f(POI_i \mid loc_n^N)$ represents the number of loc_n^N considering POI_i as the destination in Seq_u, $S(loc_n^N)$ denotes the total number of loc_n^N for different destinations in Seq_u, and $A(loc_n^N)$ describes the type set of different destinations that have the same contextual sequence with loc_n^N in Seq_u.

$$S\left(loc_n^N\right) = \sum_{i=1}^{|A(loc_n^N)|} f\left(x_i \mid loc_n^N\right), \quad xi \in A\left(loc_n^N\right) \tag{2}$$

Equation (1) shows that, given a POI_i in HPOI, if there are a sequence of trajectories with the same length as its loc_n^N, the probability p_i of POI_i that indicates POI_i would be a next POI destination that is determined and returned by $p(POI_i \mid loc_n^N)$. Otherwise, it starts to decrease N by 1 and updates the context model to be loc_n^{N-1} for predicting using $p(esc \mid loc_n^N)$. The "*esc*" in (3) indicates the escape cade which is provided in the PPM model to control the searching until it equals 0.

$$p\left(esc \mid loc_n^N\right) = \frac{f\left(esc \mid loc_n^N\right)}{S\left(loc_n^N\right) + f\left(esc \mid loc_n^N\right)} \tag{3}$$

where $p(esc \mid loc_n^N)$ indicates the prediction probability of the escape code of loc_n^N, $f(esc \mid loc_n^N)$ denotes the frequent of escape code of loc_n^N, and $f(esc \mid loc_n^N) = |A(loc_n^N)|$.

We then decrease $f(esc \mid loc_n^N)$ by 1, identify the number of loc_n^{N-1} in seq_u, and search the sequence of trajectories with the same with loc_n^{N-1} in seq_u. If there is no such sequence, continue to decrease the frequent of escape code until finding the context model loc_n^{N-r} in the r-th round. The probability is then calculated as

$$p_i = \left[\prod_{j=k+1}^{N} p\left(esc \mid loc_n^j\right)\right] . p\left(POI_i \mid loc_n^k\right), \quad k = N - r, \tag{4}$$

$$P\left(POI_i \mid loc_n^N\right) = \frac{f\left(POI_i \mid loc_n^k\right)}{S\left(loc_n^N\right) + f\left(esc \mid loc_n^N\right)}, \tag{5}$$

If no trajectory of the same with the context model is found while the frequent of escape code equals 1, we assign the prediction probability p_i as

$$p_i = \frac{1}{w'} \tag{6}$$

We can determine the type of predicted POI with the maximum probability. In the following section, social interaction has been used in conjunction with movement-related information to recommend the POIs with the same type. The similarity and relatedness between users of the same POI types are identified with social links or similar interests in the topics. The POIs with the same type from the communities' suggestions is finally provided.

3.2. Community Detection. (1) *Modeling Social Interests.* We employed the social topics of interest [30] to users to define their similarity. We define $Tweet(u)$ as a set of social-media data posted by user u. A Latent Dirichlet Allocation (LDA) [31, 32] is then applied to learn the topics of $Tweet(u)$ through word splitting, stop-word filtering, and part of speech. A vector t_u is then established for the intended topics corresponding to user u. For users u and v, the similarity of social interests is denoted $sim_{\text{interest}}(u, v)$, expressed by the following equation, where users u and v have similar interests when $sim_{\text{interest}}(u, v) \approx 1$:

$$Sim_{\text{interest}}(u, v) = \frac{\overrightarrow{t_u} . \overrightarrow{t_v}}{\left\|\overrightarrow{t_u}.\right\| \left\|\overrightarrow{t_v}\right\|}, \tag{7}$$

(2) *Modeling Travel Preference.* Our previous study suggested modeling of the *users*' movement-related information using a heterogeneous information network (HIN) [33, 34]. We identify the similarity between two users with the same travel preference in the HIN by a SimRank [35] model in a random-walk process [36].

The LBSN is first modeled as a heterogeneous information network H ⟨U, POI, E⟩, where the where the travel information in a LBSN refers to the check-in behaviors. Here, U is the set of users and POIs is the set of all POIs. $E = E_{uu} \cup E_{up} \cup E_{pp}$ denotes the set of all undirected edges in the network, where E_{uu} represents the relation between users, indicating each user pair has similar travel preferences, E_{up} implies the check-in behavior of users, and E_{pp} represents POIs of the same type. The similarity of travel between users

depends on whether two users can meet each other while randomly walking in the network H. Based on the lengths of the paths through which u and v meet and the number of times they meet, the similarity is calculated using

$$Sim_{Track}^{q+1}(u,v) = \sum_{r(u,v)\longrightarrow(x,x);l(r)\leq q+1} p[r]\, C^{l(r)}, \quad (8)$$

where $r(u,v) \longrightarrow (x,x)$ denotes the two random walks that start from u and v, respectively. Suppose that they first meet at node x in H, and the lengths of two tracks from their respective origins to x are defined as l(t). Given the two random-walk paths r_1, r_2, the probabilities of a user walking along r_1, r_2 are $P[r_1] = \prod_{i=1}^{m}(1/|O(loc_i)|)$ and $P[r_2] = \prod_{j=1}^{m}(1/|O(loc_i)|)$, respectively, where $O(loc_i)$ denotes the i^{th} nearby locations of $O(loc_i)$. The probability of u and v meeting via r_1, r_2 is then $P[r] = P[r_1].\ P[r_2] = \prod_{i,j=1}^{m}(1/|O(loc_i)\ |\ O(loc_i)|)$. To calculate $Sim_{Track}^{q+1}(u,v)$ from the random-walk perspective, all paths in H whose length is less than or equal to $q+1$ and their probability that a user walked along the paths are detected. Given a node, the similarity between two users that have the same destination and length of paths is thus determined via (8).

A direct way to combine two information sources for community discovery is to obtain a unified similarity $Sim_{Fuse}(u,v)$ by a weighted combination of all the similarity matrices as follows:

$$Sim_{Fuse}(u,v) = w_1 Sim_{interest}(u,v) + w_2 Sim_{Track}(u,v), \quad (9)$$

where $\sum_t w_l = 1, l = 1,2$, and $\forall l,\ w_l \geq 0$. We can thus obtain a set Community (u) of N users that are most similar to u.

4. Travel-Community-Based POI Recommendation

The POI recommendation algorithm proposed in this study combines the predicted POIs using personal historical trip data with the candidate POIs of the same types as those of predicted POIs generated by the detected travel communities. Such a candidate POI set L_u^* is then obtained.

To determine the potential POI locations of greatest interest to community members, it is possible to identify how a user prefers a location that can be measured by the number of times that user checks in at the given location. In general, the more a user checks in, the more he or she feels interested in the location. However, counting check-ins for a user at a given location cannot be an indicator of interest in that location because it fails to account for the number of times the user may check-in at other locations. As such, we seek to measure the relative degree of interest for a user among various POIs of the same type. Therefore, in this study we refer the degree of interest of user v on location loc_i to the proportion of the number of check-ins for v at loc_i, represented as $check\_in(v, loc_i)$, to the total number of check-ins for v at all locations of the same types as that of loc_i; it is expressed as follows:

$$Community_{interest}(v, loc_i) = \frac{check_{in}(v, loc_i) - \overline{check\_POI_{loc_i}}}{\delta_v^{loc_i}}, \quad (10)$$

where $\overline{check\_POI_{loc_i}}$ denotes the average number of check-ins of v at locations within the same types as loc_i, and $\delta_v^{loc_i}$ denotes the variance in the number of check-ins of v at locations within the same types as loc_i.

The degree of interest of user u in location $loc_i \in L_u^*$ is then expressed as

$$Interest(u, loc_i) = \frac{\sum v \in community(u)\,(sim_{Fuse(u,v)}.community_{interest}(v, loc_i))}{\sum v \in community(u)\,(Sim_{Fuse(u,v)})}, \quad (11)$$

With (7), we can provide a list of the recommended POIs from L_u^* ranked by the degree of interest of user u. Algorithm 1 recommends POIs based on *users*' historical trip data and community members who have similar social interests.

5. Results and Discussion

5.1. Dataset. We evaluated the performance of our algorithms by two check-in databases centered in New York City, i.e., Foursquare and Gowalla. Each dataset includes both check-in records and reviews. Both datasets were subjected to preprocessing, where false check-in data were removed, such that the data of a single check-in by a user during a day were collected. Then, the Foursquare dataset contained 3,357 users, 3,543 POI locations, and a total of 168,297 check-ins, whereas the Gowalla dataset contained 5,419 users, 6,742 POI locations, and a total of 330,724 check-ins.

5.2. Evaluation of Recommendation Performance. Two performance indices, denoted accuracy and recall, were used to assess the recommendation performance of the proposed TC-based POI recommendation algorithm. These indices compare distinct relationships between R(u) and the set of POI locations actually visited by a user u according to the actual check-in data (i.e., T(u)). The accuracy and recall of the POI recommendations for u are defined as follows:

$$Accuracy = \frac{\sum_{u\in U}|R(u)\cap T(u)|}{\sum_{u\in U}|R(u)|}, \quad (12)$$

$$\text{Re}call = \frac{\sum_{u\in U}|R(u)\cap T(u)|}{\sum_{u\in U}|T(u)|}, \quad (13)$$

Here, *Accuracy* reflects the accuracy of the recommendation and refers to the proportion of locations in the recommendation results that users actually visit in the future compared to the total number of POI locations recommended. *Recall* reveals the comprehensiveness of the recommendation

Table 1: Performance of LSTM-based POI recommendation algorithm.

Dataset	Metric	N=5	N=10	n=15	n=20
Foursquare	Accurary@N	22%	17%	15%	14%
	Recall@N	5%	9%	11%	12%
Gowalla	Accurary@N	23%	17%	16%	14%
	Recall@N	5%	8%	10%	12%

Input: U:all users in systems
u,v,h: users;
$POI\_category_j$: the j^{th}POI types of u
$POI_h[i]$: the i^{th} POI of h
L_u^*: set of candidate POIs
l(u): set of degree of interest of u
Output: R(u): set of POIs u is interested in
(1) begin
(2) for each $v \in U - \{u\}$ do
(3) $A = Sim_{Fuse}(u, v)$
(4) get $array[A]$
(5) end for
(6) $for(i = 0; i \leq N - 2; i + +)$
(7) $for(j = 0; j \leq N - i; j + +)$
(8) if$(array[A_j] > array[A_{j+1}])$
(9) $int\ tmp = array[A_j]; array[A_j] = array[A_{j+1}]; array[A_{j+1}] = tmp;$
(10) end if
(11) end for
(12) end for
(13) for each $h \in Community(u)$ do
(14) $for(k = 0; k \leq Y; k + +)$
(15) if $POI_h[k] == POI\_category(u)$
(16) $L_u^* = \emptyset; L_u^* = L_u^* \cap POI_h[k]$
(17) $R(u) = \emptyset; R(u) = R(u) \cap h$
(18) $I(u) = \emptyset; I(u) = I(u) \cap Interest(u, POI_h[k])$
(19) end if
(20) end for
(21) end for
(22) $R(u) = R(u) \cap Rank(I(u))$.POI
(23) return R(u)
(24) end

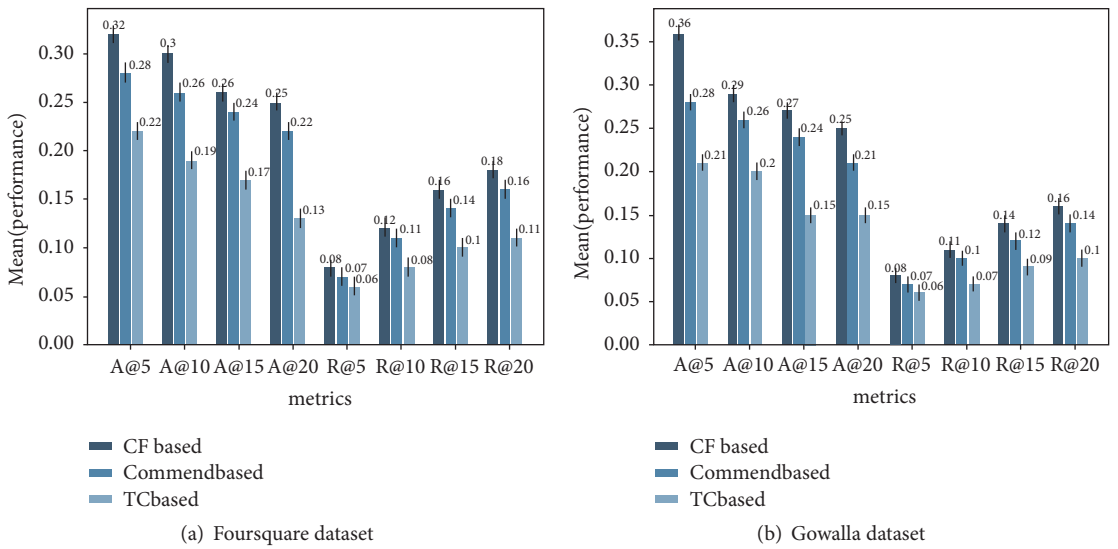
Algorithm 1: Travel-community- (TC-) based POI recommendation.

and refers to the number of locations in the recommendation results that users actually visit in the future compared to the total number of POIs that the users actually visit in the future. Accuracy and recall are mutually constrained and a comprehensive utilization of the two can provide an objective evaluation of the prediction results.

The recommendation performance using the destination prediction proposed in this study was verified by comparison with three other typical POI recommendation algorithms, in terms of accuracy and recall. The accuracy and recall of the three algorithms, in which the top-N (N=5, 10, 15, and 20) POIs are suggested, are shown in Figure 1 for the two datasets.

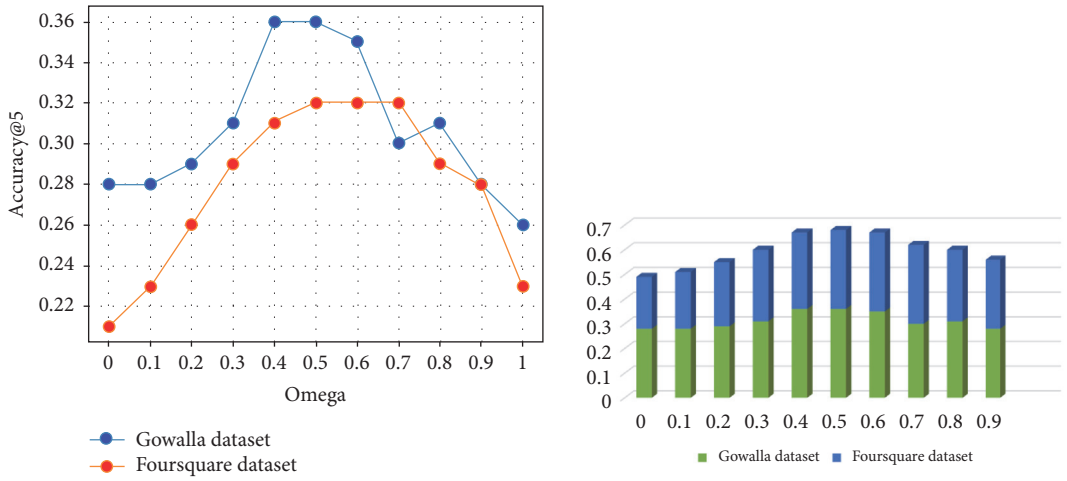
It can be seen that the TC-based POI recommendation algorithm performed better than the other two algorithms for both datasets for all values of N considered. It can be seen from the figure that our proposed algorithm achieves a better accuracy, which is 15% higher than the baseline CF-based algorithm in terms of accuracy. The performance is reasonable because the use of travel community alleviates the problem caused by data sparsity, leading to an improved POI recommendation in sparse and complex networks like LBSNs. The social interaction is also integrated into the recommendations, making the suggestions more personalized.

We also employ the LSTM (Long Short-Term Memory) algorithm to predict the destination. As illustrated in Table 1, the accuracy of LSTM-based POI recommendation algorithm is far less than that of proposed method. The reason is that the LSTM-based algorithm is likely to hinder mobility recognition without the knowledge of the latent semantic relationships between two near neighbors. Moreover, using the various travel trajectories as the input of LSTM-based



(a) Foursquare dataset (b) Gowalla dataset

Figure 1: Performance of the top-N (N = 5, 10, 15, and 20) POI recommendations.



(a) (b)

Figure 2: Choice of different combining factor between social interests and travel preference objectives.

algorithm may yield suboptimal prediction, due to the differences in the length of travel trajectories.

We study the hyperparameter w_1, which is the trade-off term for combining the interests of social and travel information. The result is shown in Figure 2 where N is 5. We use the weight $w_1 \in [0, 1]$, $w_2 = 1 - w_1$ to combine two kinds of information for fair comparison. It shows that the prediction is very low (usually less than 0.3) when $w_1 = 1$, that is, by relying on a single proximity-related metric. As shown in Figure 2(a), when we increase the weight of the social information, the performance of our algorithm will arise. But after reaching 0.5, the performance will start to go down slightly. This is because direct combination of two kinds of similarity matrices can lead to a stable community detection solution. As we can see in Figure 2(b) the best performance is obtained when we use $w_1 = w_2 = 0.5$, at which both objectives are combined most appropriately.

6. Conclusions

POI recommendations play a key role in attracting users in LBSNs. The algorithm proposed in this paper aims to optimally recommend POI types to serve multiple users. First, the intended POIs of an individual are analyzed according to their historical trip data, and a variable-order Markov model is employed to predict the types of potential POI locations for the user. Second, a degree of interest is defined to discover the community and the set of POIs according to the social links and travel preferences between users. Two types of POI information are then combined to rank the candidate POIs for a top-N recommendation. The results of experiments employing real-world datasets demonstrate that the proposed algorithm provides better accuracy and recall than two other typical POI recommendation algorithms. However, the performance of the algorithm would benefit

from further studies to model the temporal information for mining user behavior. In addition, the weighted combination in (9) would lead to limited flexibility in processing real data. Owing to the problem of community detection with multiple similarity matrices, we plan to perform multisource diffusion modeling to guarantee the maximal consistency of different data manifolds and effective information fusion.

Data Availability

The authors declare that the data supporting the findings of this study are available within the paper or from the authors upon reasonable request.

Disclosure

This work has been presented in the 2nd International Workshop on Social Computing (IWSC'18).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was funded by the NFSC under Grant 61303041, Funds of Key Scientific and Technological Innovation Team of the Shanxi Province, China, under Grant 2017KCT-29, and Funds for International Scientific and Technological Cooperation Project of the Shanxi Province under Grant 2017KW-015." Lei Tang thanks LetPub (www.letpub.com) for its linguistic assistance during the preparation of this manuscript."

References

[1] C. Cheng, H. Yang, M. R. Lyu, and I. King, "Where you like to go next: successive point-of-interest recommendation," in *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI '13)*, vol. 13, pp. 2605–2611, 2013.

[2] J. Li, C. Liu, J. X. Yu, Y. Chen, T. Sellis, and J. S. Culpepper, "Personalized influential topic search via social network summarization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 7, pp. 1820–1834, 2016.

[3] J. Chen, W. Zhang, P. Zhang, P. Ying, K. Niu, and M. Zou, "Exploiting spatial and temporal for point of interest recommendation," *Complexity*, vol. 2018, Article ID 6928605, 16 pages, 2018.

[4] A. K. M. Rahman Khan, O. Correa, E. Tanin, L. Kulik, and K. Ramamohanarao, "Ride-sharing is about agreeing on a destination," in *Proceedings of the 25th ACM Sigspatial International Conference on Advances in Geographic Information Systems*, p. 6, ACM, 2017.

[5] L. Zhang, T. Hu, Y. Min et al., "A taxi order dispatch model based on combinatorial optimization," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2151–2159, ACM, 2017.

[6] S. Wang, "A location recommendation algorithm based on location-based social networks," *Computer Engineering & Science*, pp. 458–461, 2016.

[7] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: User movement in location-based social networks," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1082–1090, San Diego, Calif, USA, 2011.

[8] S. Chen, Y. Li, W. Ren, D. Jin, and P. Hui, "Location prediction for large scale urban vehicular mobility," in *Proceedings of the 2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC '13)*, pp. 1733–1737, 2013.

[9] A. Noulas, S. Scellato, N. Lathia, and C. Mascolo, "Mining user mobility features for next place prediction in location-based services," in *Proceedings of the 12th IEEE International Conference on Data Mining (ICDM '12)*, pp. 1038–1043, IEEE, 2013.

[10] J. Li, X. Yang, X. Wang, T. Sellis, K. Deng, and J. X. Yu, "Most influential community search over large social networks," in *Proceedings of the 33rd IEEE International Conference on Data Engineering, (ICDE '17)*, pp. 871–882, 2017.

[11] M. Berlingerio, B. Ghaddar, R. Guidotti, A. Pascale, and A. Sassi, "The GRAAL of carpooling: Green and social optimization from crowd-sourced data," *Transportation Research Part C: Emerging Technologies*, vol. 80, pp. 20–36, 2017.

[12] H. Yin, B. Cui, X. Zhou, W. Wang, Z. Huang, and S. Sadiq, "Joint modeling of user check-in behaviors for real-time point-of-interest recommendation," *ACM Transactions on Information and System Security*, vol. 35, no. 2, pp. 1631–1640, 2016.

[13] J. Li, T. Cai, A. Mian, R. Li, T. Sellis, and J. X. Yu, "Holistic influence maximization for targeted advertisements in spatial social networks," in *Proceedings of the 2018 IEEE 34th International Conference on Data Engineering (ICDE '18)*, pp. 1340–1343, IEEE, 2018.

[14] J. D. Zhang and C. Y. Chow, "Point-of-interest recommendations in location-based social networks," *SIGSPATIAL Special*, vol. 7, no. 3, pp. 26–33, 2016.

[15] D. Lian, C. Zhao, X. Xie, G. Sun, E. Chen, and Y. Rui, "GeoMF: Joint geographical modeling and matrix factorization for point-of-interest recommendation," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (KDD '14)*, pp. 831–840, 2014.

[16] X. Long and J. Joshi, "A HITS-based POI recommendation algorithm for location-based social networks," in *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM '13*, pp. 642–647, 2013.

[17] Y. Zheng and X. Xie, "Learning travel recommendations from user-generated GPS traces," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 1, pp. 1–29, 2011.

[18] B. Hu, M. Jamali, and M. Ester, "Spatio-temporal topic modeling in mobile social media for location recommendation," in *Proceedings of the 13th IEEE International Conference on Data Mining, (ICDM '13)*, pp. 1073–1078, 2013.

[19] B. Liu, Y. Fu, Z. Yao, and H. Xiong, "Learning geographical preferences for point-of-interest recommendation," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '13*, pp. 1043–1051, 2013.

[20] W. Li, S.-X. Xia, F. Liu, L. Zhang, and G. Yuan, "Location prediction algorithm based on movement tendency," *Journal on Communicaitons*, vol. 35, no. 2, pp. 46–62, 2014.

[21] B. Berjani and T. Strufe, "A recommendation system for spots in location-based online social networks," in *Proceedings of the 4th Workshop on Social Network Systems (SNS '11)*, pp. 1–6, 2011.

[22] Y. Shi, P. Serdyukov, A. Hanjalic, and M. Larson, "Personalized landmark recommendation based on geotags from photo sharing sites," *ICWSM*, vol. 11, pp. 622–625, 2011.

[23] M. Ye, P. Yin, and W.-C. Lee, "Location recommendation for location-based social networks," in *Proceedings of the 18th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM SIGSPATIAL GIS '10)*, pp. 458–461, San Jose, Calif, USA, 2010.

[24] G. Ference, Y. Mao, and L. Wang-Chien, "Location recommendation for out-of-town users in location-based social networks," in *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, pp. 721–726, ACM, 2013.

[25] J. Li, T. Sellis, J. S. Culpepper, Z. He, C. Liu, and J. Wang, "Geo-social influence spanning maximization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 8, pp. 1653–1666, 2017.

[26] J. Bao, Y. Zheng, and M. F. Mokbel, "Location-based and preference-aware recommendation using sparse geo-social networking data," in *Proceedings of the 20th International Conference on Advances in Geographic Information Systems (SIGSPATIAL '12)*, pp. 199–208, ACM, 2012.

[27] M. Ye, P. Yin, W.-C. Lee, and D.-L. Lee, "Exploiting geographical influence for collaborative point-of-interest recommendation," in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '11)*, pp. 325–334, ACM, 2011.

[28] L. Chen, C. Liu, R. Zhou, J. Li, X. Yang, and B. Wang, "Maximum co-located community search in large scale social networks," *Proceedings of the VLDB Endowment*, vol. 11, no. 9, pp. 1233–1246, 2018.

[29] J. B. Clempner and A. S. Poznyak, "Multiobjective Markov chains optimization problem with strong Pareto frontier: Principles of decision making," *Expert Systems with Applications*, vol. 68, pp. 123–135, 2017.

[30] R. Guidotti and M. Berlingerio, "Where is my next friend? recommending enjoyable profiles in location based services," in *Complex Networks VII*, pp. 65–78, Springer, 2016.

[31] L. C. Lee, C. Y. Liong, and A. Z. Jemain, "Q- mode versus r-mode principal component analysis for linear discriminant analysis (LDA)," in *Proceedings of the American Institute of Physics Conference Series*, pp. 285–292, 2017.

[32] L. Qiu and J. Yu, "CLDA: An effective topic model for mining user interest preference under big data background," *Complexity*, vol. 2018, Article ID 2503816, 10 pages, 2018.

[33] C. Shi, X. Kong, Y. Huang, P. S. Yu, and B. Wu, "HeteSim: A general framework for relevance measure in heterogeneous networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 10, pp. 2479–2492, 2014.

[34] J. Wu, L. Yu, Q. Zhang et al., "Multityped community discovery in time-evolving heterogeneous information networks based on tensor decomposition," *Complexity*, vol. 2018, Article ID 9653404, 16 pages, 2018.

[35] W. Zheng, L. Zou, Y. Feng, L. Chen, and D. Zhao, "Efficient SimRank-based similarity join over large graphs," *Proceedings of the Vldb Endowment*, vol. 6, no. 7, pp. 493–504, 2013.

[36] H. Gao, J. Tang, and H. Liu, "Personalized location recommendation on location-based social networks," in *Proceedings of the 8th ACM Conference on Recommender Systems, RecSys '14*, pp. 399-400, 2014.

Hindawi
Complexity
Volume 2019, Article ID 4042624, 14 pages
https://doi.org/10.1155/2019/4042624

Research Article

A Block Object Detection Method Based on Feature Fusion Networks for Autonomous Vehicles

Qiao Meng,[1,2] **Huansheng Song**,[1] **Gang Li**,[1] **Yu'an Zhang**,[2] **and Xiangqing Zhang**[1]

[1]*School of Information Engineering, Chang'an University, Xi'an, Shaanxi 710064, China*
[2]*Computer Technology and Application Department, Qinghai University, Xining 810016, China*

Correspondence should be addressed to Qiao Meng; 250345481@qq.com

Received 24 November 2018; Accepted 21 January 2019; Published 6 February 2019

Guest Editor: Ke Deng

Nowadays, automatic multi-objective detection remains a challenging problem for autonomous vehicle technologies. In the past decades, deep learning has been demonstrated successful for multi-objective detection, such as the Single Shot Multibox Detector (SSD) model. The current trend is to train the deep Convolutional Neural Networks (CNNs) with online autonomous vehicle datasets. However, network performance usually degrades when small objects are detected. Moreover, the existing autonomous vehicle datasets could not meet the need for domestic traffic environment. To improve the detection performance of small objects and ensure the validity of the dataset, we propose a new method. Specifically, the original images are divided into blocks as input to a VGG-16 network which add the feature map fusion after CNNs. Moreover, the image pyramid is built to project all the blocks detection results at the original objects size as much as possible. In addition to improving the detection method, a new autonomous driving vehicle dataset is created, in which the object categories and labelling criteria are defined, and a data augmentation method is proposed. The experimental results on the new datasets show that the performance of the proposed method is greatly improved, especially for small objects detection in large image. Moreover, the proposed method is adaptive to complex climatic conditions and contributes a lot for autonomous vehicle perception and planning.

1. Introduction

Environment perception is an important part of the autonomous driving system, and the sensors used for sensing include ultrasonic radar, millimetre wave radar, LiDAR (Light Detection and Ranging), and cameras. Through the fusion of LiDAR, millimetre wave radar, and cameras, objects can be detected, and object space ranging and recognition can be realized. Specially, the fusion of cameras and LiDAR not only can realize high precision positioning of objects but also can realize the detection of multiple types of objects. However, due to the high cost of LiDAR, this sensor fusion cannot become a popular method in the future. In contrast, low-cost cameras will be applied in the autonomous vehicle perception system, such as object detection and classification.

Currently, there are two object detection algorithms, namely traditional image processing and deep learning classification. Through the analysis and processing of images, both methods can return the location and classification information of objects and provide effective information for the planning and decision-making system. However, due to the extremely rich information of images and the difficulty in manual modelling, the accuracy of the traditional image processing method is worse than the deep learning method. Therefore, more and more camera-based deep learning algorithmscan make the perception of autonomous vehicles much more accurate, fast and comprehensive. At present, the existing deep learning systems can be divided into two categories. One is the region proposal method, such as R-CNN [1], Fast R-CNN [2] and Faster R-CNN [3]. The other is the proposal-free method, such as You Only Look Once (YOLO) [4] and Single Shot Multi-box Detector (SSD) [5]. In recent years, SSD model has obvious advantages for video object detection in terms of detection speed and accuracy. However, some problems still exist in the SSD model. The first problem is the dataset. A rich dataset is crucial for object detection. At present, the current datasets for autonomous driving are based on foreign traffic scenarios, such as KITTI, Cityscapes, etc. The second problem lies in the classification accuracy. The detection accuracy of SSD model is lower

Table 1: Compare the performance of different SSD512 models tested on the KITTI dataset.

Structure	Type	KITTI test (mAP)	KITTI test (fps)
SSD512	No feature map fusion	75.6	42
DSSD512	Merging feature map including extract different scales	76.8	31
Our SSD512	Merging feature map before extract different scales	77.4	35

than Faster R-CNN. Specially, as the network deepens, small objects are gradually lost during the convolution process of the SSD model.

In this study, we propose a novel approach to detect the object for autonomous driving. Our contribution of this paper includes four points. Firstly, we divide the original image into blocks (the block size is 400×400), which can detect small objects from the image, and then we resize each block to a fixed size (512×512) for training. Secondly, the original image is down-sampled in multiples of 1/2 times until the image size is close to the block size, which ensures that large objects can be completely covered in a single image block. Thirdly, as the feature map of the SSD model gradually shrinks, the characteristic information of the small objects disappears or becomes inconspicuous. Therefor, a feature fusion method is added in the SSD model to ensure the detection precision of small objects in the large image. Fourthly, the samples are collected and labelled by ourselves, where we have designed object categories and the annotation method.

This paper will be described as follows: Section 2 introduces our method in detail. Section 3 describes the experiment of our method. Section 4 shows the experimental results and analyzes the experimental results. Section 5 provides a discussion and the future work for this paper.

2. Methodology

In this section, we described the details of the improved SSD model using feature fusion and image block segmentation methods, and introduced the method for creating an autonomous driving dataset.

2.1. Feature Fusion Network. The SSD model directly extracts different scales from different feature map layers of the CNNs, as shown in Figure 1(a). This approach cannot fuse feature maps of different scales, so the feature maps of different scales are independent of each other. Therefore, based on the SSD model, we propose a new image feature fusion algorithm, which requires multiple feature merging processes and usually consumes so much time.

As shown in Figure 1(b), after VGG16 network, seven convolution layers were added to extract features, including conv6_1, conv6_2, conv7_2, conv8_2, conv10_2, and conv11_2. The feature map sizes of these seven convolution layers are 32×32, 32×32, 16×16, 8×8, 4×4, 2×2, 1×1. Through analysis, when the feature map size is less than 16×16, objects continue to shrink, and its characteristics gradually disappear, so feature fusion cannot be implemented. In this paper, the bilinear function is used to fuse feature maps of different sizes. The feature map is upsampled starting from conv7_2 and its size is not less than 16×16. Since the bilinear function converts one pixel to four pixels, that is, the current image is doubled, therefore, conv7_2, conv6_2, and conv4_3 satisfying the 2-fold relationship are selected for upsampling. Before the bilinear operation, a conv 1×1 operation is performed on the image, whichcan reduce the dimension of the featureand accelerate the computation [6]. The specific calculation method is as follows:

$$Y_i = Bilinear\left(y_i\right) \tag{1}$$

where y_i means a feature map that needs to be merged, *Bilinear* is linear interpolation function, and Y_i is a feature map that is magnified double. Through the calculation of this equation, the fused feature maps are adjusted to the same size. Moreover, the feature maps of the same size are fused using the element-wise-max method, which preserves the maximum value of the corresponding position pixel values in the two feature maps. The specific calculation method is as follows:

$$element - wise - \max\left(Y_i, Y_j\right) = T\left(\max\left(X_{i,j}, Y_{i,j}\right)\right) \tag{2}$$

where Y_i, Y_j represent feature maps with the same size, T represents a new feature map matrix generated after fusion, and $X_{i,j}, Y_{i,j}$, respectively, represent the pixel values of the merged feature map matrix. Through the feature fusion of the element-wise-max function, the pixel layer is generated, and the pixel layer is continuously sampled to generate a pyramid feature map, this is the feature fusion method of this paper. Table 1 compares the mean average precision (mAP) and the frame rate (fps) for different SSD models.

2.2. Dataset. For deep learning, the importance of datasets is unquestionable. A rich dataset is crucial for object detection. Currently, datasets in the field of computer vision include ImageNet [7], COCO [8] and PASCAL VOC [9], etc. Moreover, the dataset for autonomous driving primarily uses the KITTI [10] dataset or the Cityscapes [11] dataset. Table 2 compares the two datasets.

Since the KITTI or Cityscapes dataset for autonomous driving satisfies the requirements of computer visual, the perspective and categories of samples do not match well with domestic demand. Therefore, it is necessary to establish a new dataset which not only satisfy the perspective requirements but also match the current domestic traffic environment well.

Table 2: Comparison of different datasets.

Dataset	Categories	Function	Time
KITTI	Car, Van, Truck, Pedestrian, cyclist, Tram	Evaluation of computer vision algorithm in automatic driving scene	2012
Cityscapes	Flat, Nature, Vehicle, Sky, Object, Human, Construction	Semantic urban scene understanding	2016

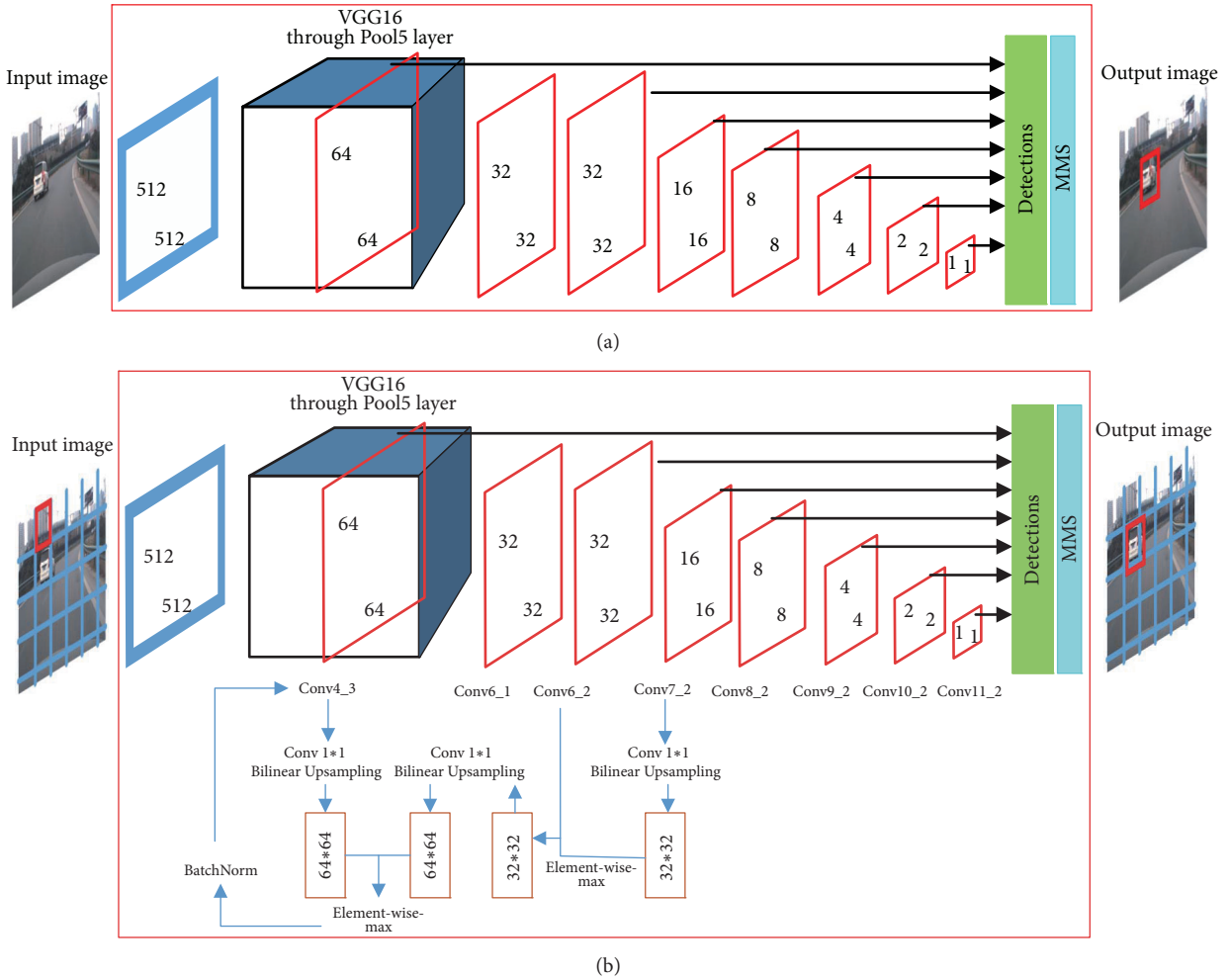


Figure 1: Our SSD512 framework. The part of the red box labelled in figure (a) is the SSD framework proposed in [5], and the part of the red box labelled in figure (b) is our SSD512 framework. Figure (b) uses a pyramid feature fusion method that fuses feature maps with a 2-fold relationship in a recursive manner.

2.2.1. Sample Collection

(1) Sample Collection Platform. In order to create a new dataset that meets our requirements, we apply a data acquisition platform for autonomous vehicles to collect samples from real traffic environments. The data acquisition platform is equipped with a high-dynamic camera (acA1920), a velodyne 64-E LiDAR, and a differential GNSS receiver (Simpeak 982). The dataset we have collected contains real-world image data from urban, rural, freeway scenarios, etc. Moreover, each image contains at least one vehicle or one pedestrian. The entire system sampled and synchronized at 10Hz frequency. In the image acquisition system, the cameras shooting distance of our autonomous vehicle is 13 meters, and the resolution of the captured image is 1920×1200. Moreover, referring to the current status of China's traffic roads, we classify the labels into seven categories, including car, truck, bus, minibus, cyclist, person, and motorcycle.

(2) Sample Augmentation. Traffic scenes captured by image acquisition platforms include urban roads, highways, tunnels,

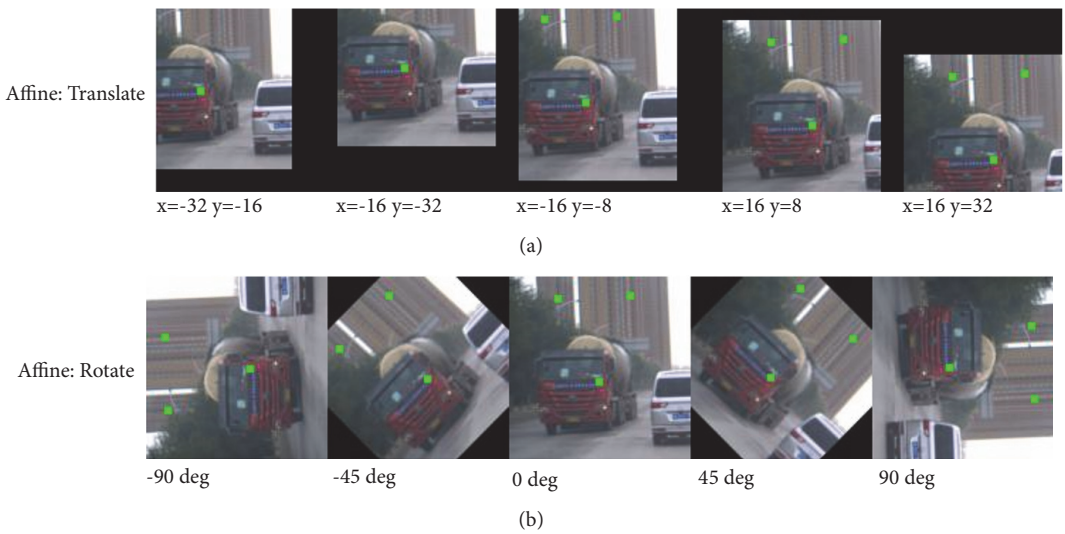


Figure 2: The example image represents how the augmentation method "sees" objects. (a) Shift the image according to the coordinate system. (b) Rotate the image at different angles.

and curved roads. Moreover, some samples were collected under complex climatic conditions, such as rainy days and foggy days. Since deep learning requires training a large number of samples to learn object characteristics well, we use sample augmentation method to expand the dataset. As shown in Figure 2, the specific process is as follows.

Step 1. Randomly select an image from all the pre-trained images.

Step 2. For each image, one of the small blocks is randomly sampled, the aspect ratio of the small block is set to$[1/2, 2]$, and the overlap ratio with the object is 0.1, 0.3, 0.5, 0.7, and 0.9.

Step 3. If the central point of the bounding box is in the sampled block, the overlapped portion is retained.

Step 4. This article uses a fixed size of 512×512. To resize each sample to the fixed size, and then shifts or rotates the fixed block at a random level with a probability of 0.5.

2.2.2. Annotation. The diversity of samples is crucial to ensure the accuracy of detection [12]. Therefore, the selection of samples should consider multiple angles, complex climatic conditions, occlusion ratio and truncation ratio. The principles we propose for image annotation are as follows.

(1) Select Samples from Multiple Angles. There are slight differences in the characteristics of the samples from different angles [13]. Therefore, angles are crucial for the image. We label samples from the positive angle, the reverse angle, and the side angle to ensure the comprehensiveness of samples. Table 3 shows the statistics for the number of seven types of samples from different angles.

(2) Choose Complex Climate Conditions. In severe weather conditions, the model is greatly affected by visibility. After analyzing the acquisition height and clarity of the cameras, we divided the bad weather into fog, rain, sunny, and cloudy. Moreover, we label samples in different climate conditions to ensure the adaptability of sample characteristics to the environment.

(3) Set the Occlusion Ratio. The human eye can easily follow a specific object for a period time [14]. However, for the machine, this task is not simple. Generally, there are various complicated situations in the object tracking process, such as the occlusion ratio is an important problem. After investigation, we define a score for an occluded bounding box. The specific definition method is divided into three cases: heavy occlusion, partial occlusion and no occlusion. In this article, if the occlusion ratio of a vehicle is larger than 40%, we define it as the heavy occlusion. Similarly, when the occlusion ratio of a vehicle is between 1% and 40%, we think it is partial occlusion. In order to ensure the accuracy of the detection, we stipulate that only partial occlusion and no occlusion are labelled, and heavy occlusion is not labelled.

(4) Set Truncation Ratio. Not all objects in the image are labelled. According to the requirements of network training, the truncation ratio is set to 1/3. In other words, if the area of the object beyond the image boundary is greater than 1/3 of the object area, then we do not label this object.

According to the defined labelling principle, the specific labelling process includes three steps. The first step is to determine the storage location of the labelled file and the storage location of the newly created dataset, and draw a bounding box for each object in the sample. The second step is to assign a label category to each object's bounding box. The third step is to determine whether the object in the sample is occluded or truncated, if it exists, a description field needs to be added for the object label.

Table 3: Samples statistic of SSMCAR datasets from different angles.

	Positive angle				Reverse angle				Side angle			
	fog	rain	sunny	cloudy	fog	rain	sunny	cloudy	fog	rain	sunny	cloudy
Person	187	94	385	201	109	81	175	184	176	79	293	183
Cyclist	133	35	257	123	36	47	214	169	73	41	121	144
Motorcycle	89	47	101	82	59	42	97	59	33	29	61	64
Car	198	241	278	261	142	169	205	193	192	105	217	254
Bus(MiniBus)	106	166	183	201	137	121	156	75	143	152	224	104
Bus	99	102	129	79	101	66	98	106	136	97	165	144
Truck	129	89	153	128	96	108	124	152	185	123	148	267

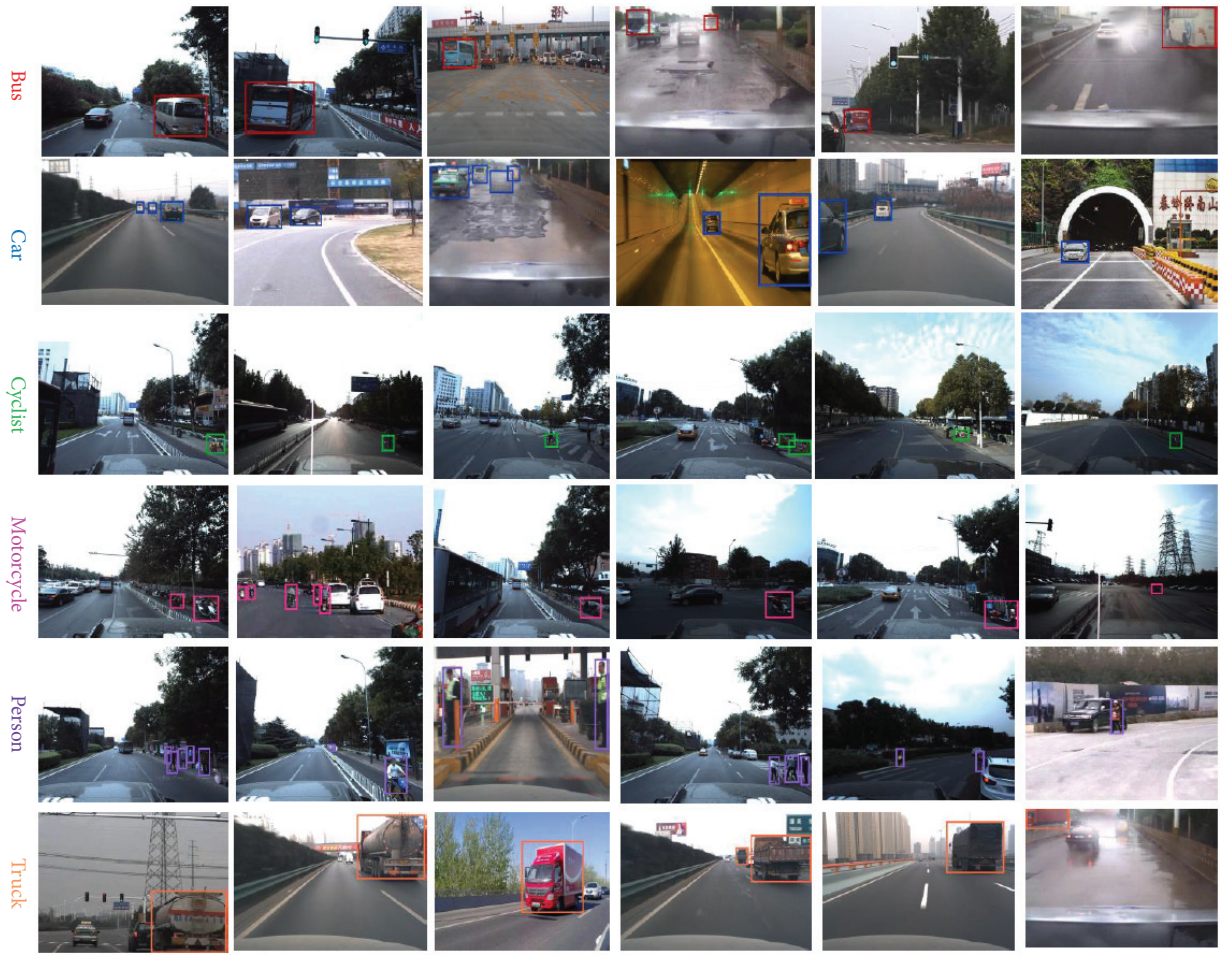


Figure 3: Snapshots are the seven categories of objects and their notations built by ourselves. Among them, the bus category in the figure represents bus and minibus. Moreover, this dataset will be used in further experiment.

In this work, a total of 11550 images are labelled, including 10394 training sets and 1156 testing sets. Moreover, the new dataset is named SSMCAR. Figure 3 is a snapshot of the SSMCAR dataset annotation.

2.3. Image Block Architecture. From the limited available memory of current GPUs, it is not feasible for deep convolutional networks to accept large images as input, especially for image sizes larger than 2000×2000 [14]. In the SSD detection model, it resized the entire image to a fixed size. As shown in Figure 1(a), it resizes the image to 512×512. The disadvantage of this approach is that it directly resizes the image to a fixed size, which not only reduces the resolution of the image itself but also affects the learning effectiveness of the object characteristics, especially for large images. Therefore, an object detection method based on image blocks is proposed. As shown in Figure 1(b), the input image is divided into blocks according to a certain strategy [15], and then each block is trained according to our SSD method.

In our SSD512 framework, in order not to change the quality of the image itself to the greatest extent, we propose a strategy to divide the original image into blocks with different sizes. Since the SSD model needs to resize the image to a fixed size in advance, such as resizing the image size to 300×300 or 512×512. At the same time, the characteristics of the convolutional neural network show that the minimal adjustment of the original image has a small influence on the final detection results. Therefore, we use the enumeration method to divide the original image into blocks around the size of 300×300 or 512×512, and then choose the best block scheme. After the image block is completed, each block will be resized to 300×300 or 512×512 before they are input into our SSD model. This approach has two advantages. On the one hand, it reduces the loss of small objects in the process of network learning, on the other hand, it reduces the problem of image quality degradation in the SSD method. Admittedly, the larger the size of the image, the better the detection result of this method. In this paper, the main purpose of our research is multi-object detection for automatic driving. Therefore, we use our own dataset as an example to illustrate the specific details of the block strategy.

In this paper, the size of the sample is 1920×1200. We assume that the fixed size of the input network is 512×512, and use the enumeration method to select different sizes close

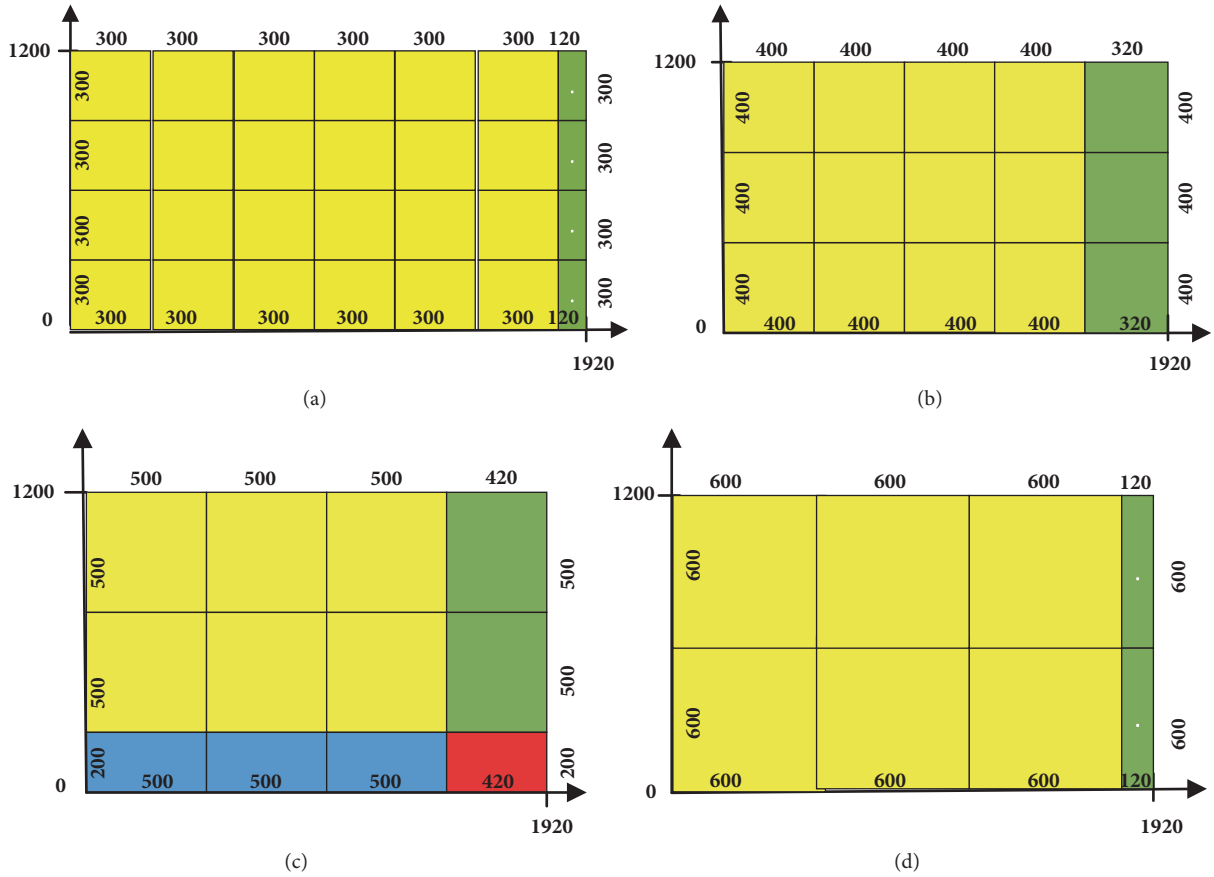


Figure 4: Different blocking strategies for 1920×1200 images. The same color represents blocks of the same size, and different colors represent blocks of different sizes. The 300×300 block strategy yields two different sizes of 300×300 and 120×300 blocks, including 24 blocks of 300×300 and 4 blocks of 120×300, as depicted in (a). Similarly, the 400×400 block strategy yields two different sizes of 400×400 and 320×400 blocks, including 12 blocks of 400×400 and 3 blocks of 320×400, as depicted in (b). Moreover, the block strategies of 500×500 and 600×600 are the same as those of Figures (a) and (b), as shown in (c) and (d).

to 512×512 to segment the original image. The calculation formula is shown as follows:

$$(x, y) = ((300, 300), (400, 400), (500, 500), (600, 600)) \quad (3)$$
$$x, y \in [300, 700)$$

where x, y represent the horizontal and vertical block sizes. As shown in Figure 4, using the top-to-bottom and the left-to-right method divides the image along the horizontal and vertical directions.

Since the network model we used in Figure 1(b) is a 512×512 model, we need to resize the block to the size of 512×512 before sending it to the network. We define two criteria to choose the best block scheme. One is that the size of the block is the closest to 512×512; the other is that the difference between blocks and blocks is the smallest and the aspect ratio of each block is the largest. In Figure 4(b), the 400×400 block strategy yields two different sizes of 400×400 and 320×400 blocks, including 12 blocks of 400×400 and 3 blocks of 320×400, based on our defined blocking scheme, we find that Figure 4(b) is the best blocking strategy which will produce the best learning effect and minimum error for our SSD model. So this paper uses this blocking method, which divides the image into 400×400 blocks.

3. Experiment

3.1. Training of Our SSD Model. Our SSD model has a large training parameter. If we train all the characteristics of the network from scratch, it is not only time-consuming but also prone to data overfitting and gradient non-convergence [15]. In this paper, the transfer learning [16] method is applied. Based on the pre-trained model, the training accuracy and loss function of the model are compared using different datasets and different network.

During the training process, our SSD model takes all the anchors in each block in a graph as the window to determine whether there is an object in the window. If there is an object, it predicts the category and position information of the object, otherwise, it defines the anchor as the background. As shown in Figure 5, our SSD model cuts each block of the graph into 8732 anchors of different sizes, each of which has

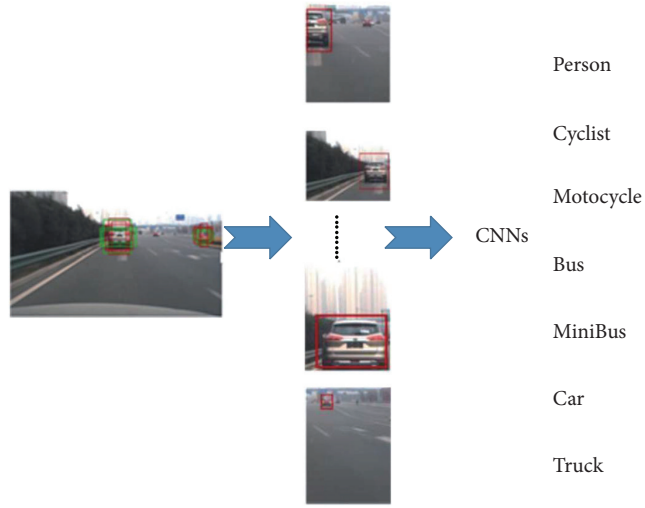


Figure 5: Training method of our SSD model. Left: a 400×400 block that is resized to 512×512. Middle: divided the graph into anchors of different sizes. Right: our SSD model trains these anchors to predict the category and location of objects.

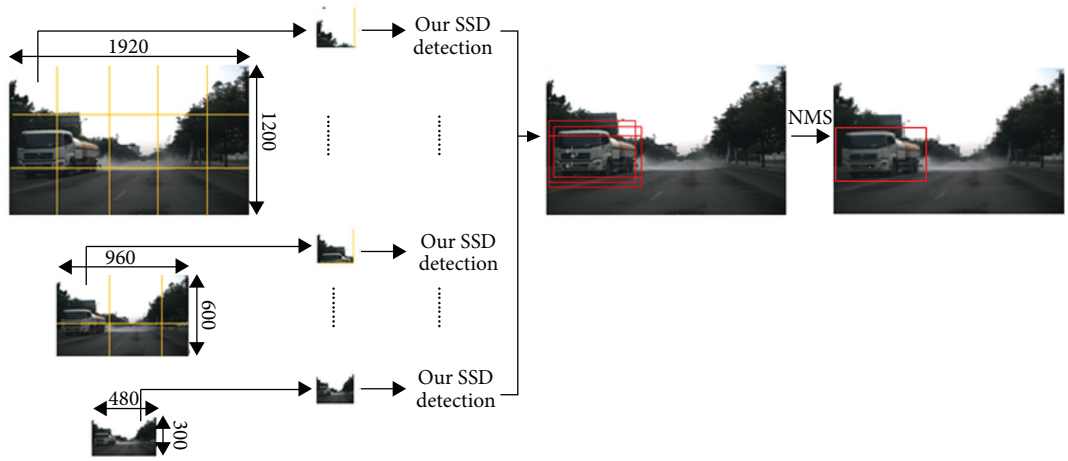


Figure 6: An illustration of proposed image pyramid architecture. The original image is divided into blocks as input to our SSD model to produce object detection result. In addition, an image pyramid is established. Each layer of image can predict an object detection result. Therefore, the non-maximal suppression is adopted to generate the final detection results.

a region of interest (ROI) [17]. We used those 8732 anchors as a batch to train, thus, we define that if the ROI and the object satisfy the condition that the overlap ratio is greater than 0.7, the label is set as the object and the offset of the region from the corresponding anchor is predicted, otherwise, the label is set as the background.

3.2. Loss Function of Our SSD Model. The loss function is applied to evaluate the network performance of our SSD model [18]. The loss function is divided into the localization loss (loc) and classification loss (conf) [19, 20], which is defined as follows:

$$L(x, c, l, g) = \frac{1}{N}\left(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)\right) \quad (4)$$

where x is a matched default box; N is all matched default boxes, if $N = 0$, then $Loss = 0$. L_{conf} is the softmax loss over object classes which is actually the loss of confidence [5]; L_{loc} is the Smooth_{L1} loss [5] based on the predicted box; α is set to 1 by cross validation.

3.3. Test Our SSD Model

3.3.1. Image Pyramid Architecture. As our SSD model is designed to be sensitive to small objects, some large objects are divided into different blocks, which can cause the loss of features of large objects at the original resolution. An image pyramid is created to solve this problem. Specifically, an image pyramid rule for constructing image pyramid is proposed, in which the size of low-resolution image are 0.5 times than the size of high-resolution image. Moreover, since our network model is 512 × 512, see Figure 1(b), if the image size is less than 512 × 512 too much, it will be detrimental to the learning of object characteristics. As shown in Figure 6,

the resolution of our dataset image is 1920×1200, and the image size of the third layer of the image pyramid is 480×300. According to our pyramid construction method, the image size of the fourth layer is 240×150 which is no value in learning object features, so a three-layer image pyramid structure is constructed.

3.3.2. Non-Maximum Suppression Method. We noticed that our SSD model predicts a result for each layer of images. As shown in Figure 6, we can see that there are three bounding boxes in the same location in our image pyramid architecture, which can generate non-uniqueness of detection. Therefore, the NMS [21, 22] algorithm can be used to eliminate redundant (cross-repeat) windows and find the best object detection location.

4. Results and Analysis

There are three parameters for evaluating object detection performance, including accuracy, loss, and detection rate [23–26]. Specifically, the parameter mAP is the average classification accuracy of the seven objects, and its value is between 0 and 1. The larger the value of mAP, the higher the classification accuracy. In addition, the frame rate (fps) is used to evaluate the detection speed.

4.1. Accuracy. In this paper, we, used SSD model and our SSD model to train and test VOC dataset, KITTI dataset, and our dataset (SSMCAR). In this process, the single NVIDIA 1080Ti GPU server is applied, the initial learning rate is set to 0.01, and the number of iterations is set to 100,000 and 120,000 times.

As shown in the left of Figure 7, the accuracy of the SSMCAR dataset is much higher than the VOC2007 dataset or KITTI dataset. Since the resolutions of the VOC2007 dataset and KITTI dataset are 500×375 and 1242×375, correspondingly, the resolution of the SSMCAR dataset is 1920×1200, the image quality of the SSMCAR dataset is higher than that of the VOC2007 dataset and the KITTI dataset. In addition, the collection perspective and the labelling principles of SSMCAR datasets are different from VOC2007 dataset and KITTI dataset. After further experiments, it shows that increasing the number of samples has no effects on improving the accuracy. Moreover, from Figures 7(a) and 7(b), it can be seen that the optimal solution occurs when the number of iterations of VOC dataset and KITTI dataset reaches 100,000 times and the highest accuracy is obtained. As shown in Figure 7(c), the optimal solution for the SSMCAR dataset we created appeared between 100,000 and 120,000 iterations.

As shown in the right of Figure 7, the accuracy of our SSD model is higher than the SSD model. Unlike the SSD model directly resize images to 512×512, our SSD model use the image block, image pyramid, and feature fusion method to protect the feature of the original images.

4.2. Loss. After each complete training process, a suitable learning rate can guarantee that the loss is reduced to a small value after a period of time [27, 28]. Too small learning rate often makes the loss reduction very slow. Conversely, if the learning rate is set too large, the initial loss can be reduced very fast, and then it repeats at a certain distance from the minimum loss value without falling [29]. Therefore, the initial learning rate is set to 0.01, and the learning rate decreases with each iteration. Moreover, loss is the total loss; it includes classification loss, localization loss and object detection loss. In this paper, as the training time increases, the total loss gradually decreases until it becomes stable and the training reaches convergence; otherwise, if the training continues when the training reached convergence, overfitting will occur.

As shown in Figure 8, our SSD model has longer training time than the SSD model, but its total loss value is the smallest. Moreover, the comprehensive performance of loss and time for our datasets is superior to the VOC2007 dataset and KITTI dataset.

4.3. Detection Rate. Randomly capture autonomous driving videos in different traffic scenarios to test the results of the objects detection. As shown in Figure 9, under the same detection model, the performance of our dataset is better than the KITTI dataset. Meanwhile, under the same dataset, the detection result of our SSD model is far superior to the SSD model. Especially, whether the object is small or large, our method can effectively detect and classify objects. Moreover, our approach is well adapted to environment and climate changes. At the same time, the statistics on Table 4 shows the classification detection accuracy, average detection accuracy, and detection rate of different methods.

5. Conclusions and Future Work

In this study, we propose an objects detection algorithm for autonomous driving based on SSD model. Through the feature fusion of the convolution layers, the effective transmission of the object features is guaranteed. In the process of training, a strategy of image block method was added to improve the detection performance of small objects. Moreover, we propose an image pyramid for big objects, which effectively solves the problem of large objects feature loss caused by image segmentation. However, in this paper, we defined labelling criteria and object categories to create a new dataset for autonomous vehicle technologies. Our experimental results show that the proposed detection algorithm for autonomous driving has good detection performance in this paper.

In future work, we will track the objects based on the results which have been detected, and then analyze the motion trend of the object to provide effective support for decision-making and path planning of the autonomous vehicles [30].

Data Availability

Data availability in this article is true and reliable.

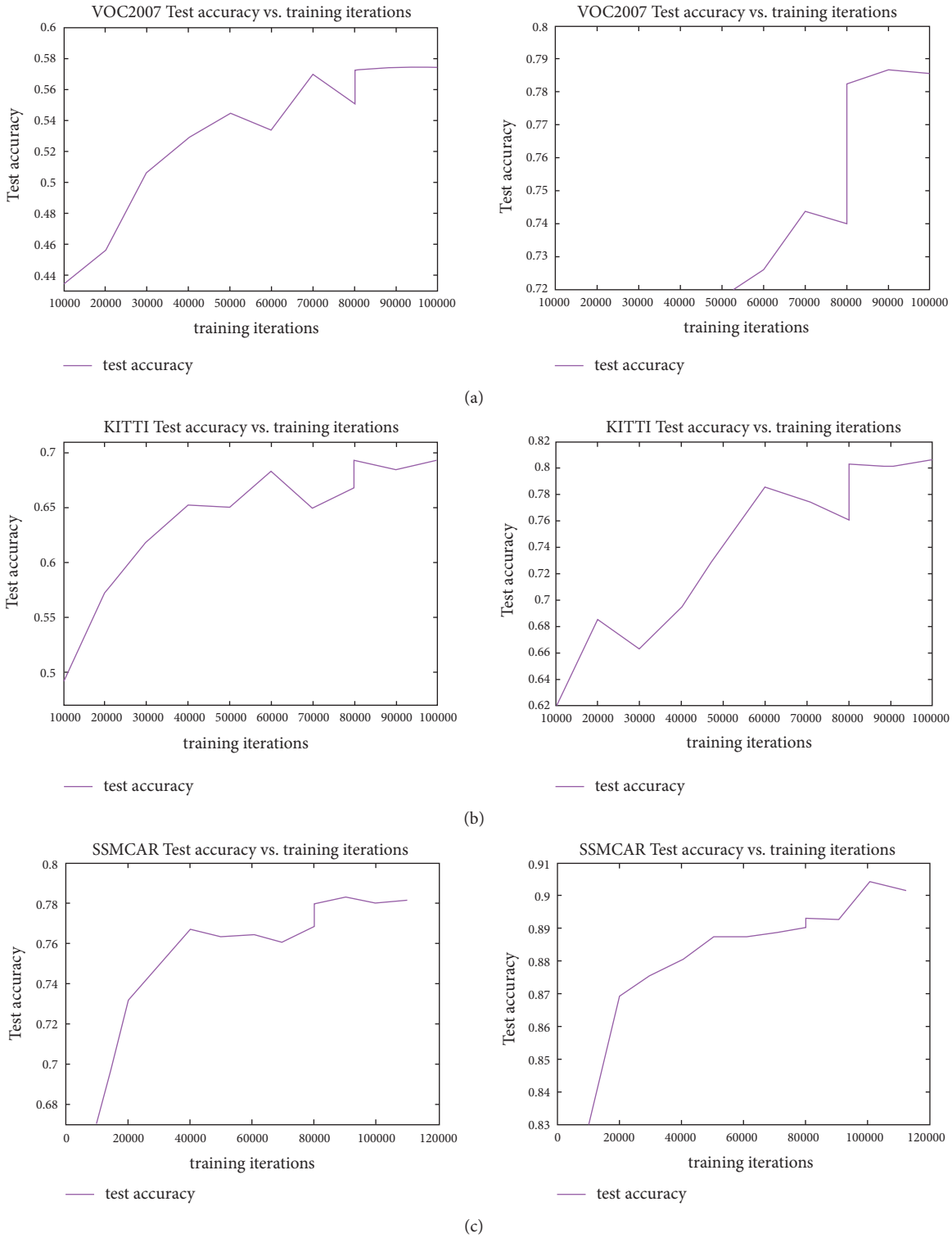


Figure 7: An illustration of the accuracy curve during training and testing stage. The abscissa expresses the value of the accuracy (%), and the ordinate expresses the number of iterations (times). Left: the accuracy curve of the SSD model on the VOC2007 dataset, KITTI dataset, and SSMCAR dataset. Right: the accuracy curve of our SSD model on the VOC2007 dataset, KITTI dataset, and SSMCAR dataset.

Table 4: Comparison of object detection effects on different models.

Detection model	Data set	Number of categories	Total number of training samples	Number of verification samples	Number of test sets	Detection accuracy of each category							Total accuracy	Detection rate
						Person	Cyclist	Motocycle	Car	MiniBus	Bus	Truck		
SSD	VOC-2007	20	9963	5011	4952	0.5145	0.5379	0.5701	0.5883	0.5534	0.5499	0.5407	0.5507	28Fps
	KITTI	5	7982	7183	799	0.6145	0.6379	0.6701	0.6883	0.6534	0.6499	0.6407	0.6507	24Fps
	SSM-CAR	7	11550	10394	1156	0.7903	0.8057	0.8109	0.8044	0.8170	0.7968	0.8102	0.8050	27Fps
Our SSD	VOC-2007	20	9963	5011	4952	0.7741	0.7856	0.7812	0.7821	0.7830	0.7904	0.7923	0.7841	31Fps
	KITTI	5	7982	7183	799	0.8145	0.8279	0.8001	0.8083	0.8234	0.8499	0.8407	0.8107	29Fps
	SSM-CAR	7	11550	10394	1156	0.9805	0.8972	0.9196	0.9585	0.9043	0.8807	0.8984	0.9085	33Fps

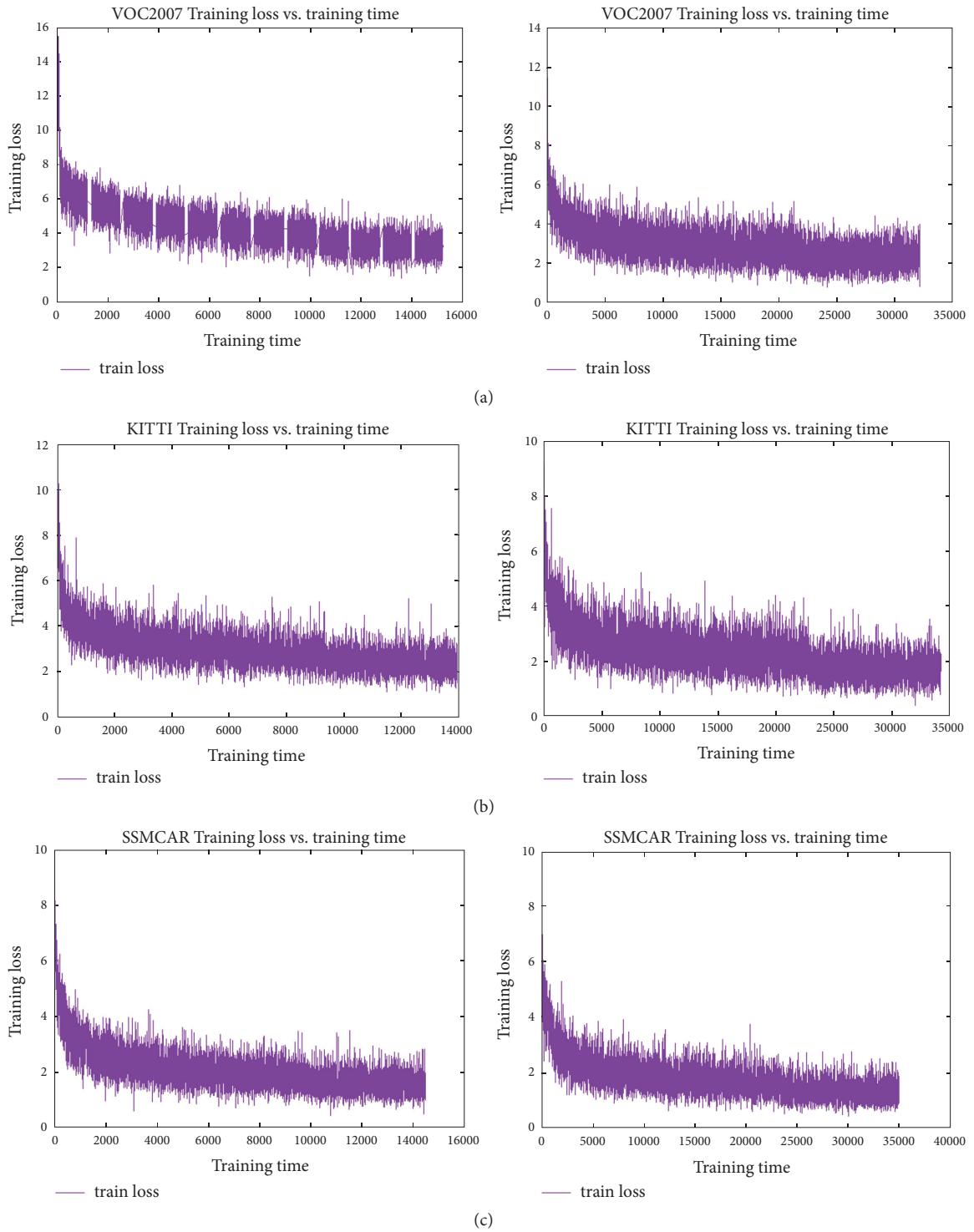


Figure 8: An illustration of the total loss curve which changes with the training time during training stage. The abscissa expresses the value of the total loss, and the ordinate expresses the value of the time. Left: the total loss curve of the SSD model on the VOC2007, KITTI, and SSMCAR datasets. Right: the total loss curve of our SSD model on the VOC2007, KITTI, and SSMCAR datasets.

Conflicts of Interest

There are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is funded by the National Natural Science Foundation of China (Grant no. 61572083), the Project of Shaanxi

FIGURE 9: The result of the objects detection for autonomous driving. Left: it is the detection result of SSD+KITTI. Middle: it is the detection result of SSD+SSMCAR. Right: it is the detection result of our SSD+SSMCAR. (a) is the detection result of urban roads. (b) is the detection result of the intersection. (c) is the detection result of the highway.

Provincial Science and Technology Program (Grant no. 2014JM8351), and the Science and Technology Project in Qinghai Province (no. 2017-ZJ-717).

References

[1] G. Ross, D. Jeff, D. Trevor, and M. Jitendra, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Computer Vision and Pattern Recognition*, vol. 1, 2014.

[2] R. Girshick, "Fast R-CNN," in *Proceedings of the 15th IEEE International Conference on Computer Vision (ICCV '15)*, pp. 1440–1448, December 2015.

[3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," in *NIPS*, 2015.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 779–788, July 2016.

[5] W. Liu, D. Anguelov, D. Erhan et al., "SSD: single shot multibox detector," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 9905, pp. 21–37, 2016.

[6] K. Deng, J. X. Li, J. Y. Li, C. Y. Pang, and X. F. Zhou, "Access time oracle on weighted planar graph," *IEEE Transactions on Knowledge and Data Engineering TKDE*, vol. 28, no. 8, pp. 1959–1970, 2016.

[7] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," in *ICLR*, 2014.

[8] J. Hosang, R. Benenson, P. Dollar, and B. Schiele, "What makes for effective detection proposals?" *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 4, pp. 814–830, 2016.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[10] G. Wang, X. Wang, B. Fan, and C. Pan, "Feature extraction by rotation-invariant matrix representation for object detection in aerial image," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 6, pp. 851–855, 2017.

[11] Y. Jia, E. Shelhamer, J. Donahue et al., "Caffe: convolutional architecture for fast feature embedding," in *Proceedings of the ACM Conference on Multimedia (MM '14)*, pp. 675–678, ACM, Orlando, Fla, USA, November 2014.

[12] M. Everingham, L. van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[13] J. Li, C. Liu, and J. X. Yu, "Context-based XML keyword query diversification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 3, pp. 660–672, 2015.

[14] Y. Lu, W. Wang, J. Li, and C. Liu, "XClean: Providing valid spelling suggestions for XML keyword queries," in *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering, ICDE 2011*, pp. 661–672, Germany, April 2011.

[15] N. Alduaiji, A. Datta, and J. Li, "Influence propagation model for clique-based community detection in social networks," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 2, pp. 563–575, 2018.

[16] Y. Zhu, R. Mottaghi, E. Kolve et al., "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *Proceedings of the 2017 IEEE International Conference on Robotics and Automation, ICRA 2017*, pp. 3357–3364, Singapore, June 2017.

[17] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "DeepDriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of the 15th IEEE International Conference on Computer Vision, ICCV 2015*, pp. 2722–2730, Chile, December 2015.

[18] L. D. Jackel, D. Sharman, C. E. Stenard, B. I. Strom, and D. Zuckert, "Optical character recognition for self-service banking," *AT&T Technical Journal*, vol. 74, no. 4, pp. 16–24, 1995.

[19] W. Danwei and Q. Feng, "Trajectory planning for a four-wheel-steering vehicle," in *Proceedings of the 2001 IEEE International Conference on Robotics & Automation*, pp. 21–26, 2001.

[20] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[21] A. Zweig and D. Weinshall, "Exploiting object hierarchy: combining models from different category levels," in *Proceedings of the 2007 IEEE 11th International Conference on Computer Vision*, pp. 1–8, Rio de Janeiro, Brazil, October 2007.

[22] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognition Letters*, vol. 30, no. 2, pp. 88–97, 2009.

[23] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *Proceedings of the 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 951–958, USA, June 2009.

[24] M. M. Anwar, C. Liu, and J. Li, "Discovering and tracking query oriented active online social groups in dynamic information network," *World Wide Web*, 2018.

[25] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Advances in Neural Information Processing Systems*, vol. 26, NIPS, 2013.

[26] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time-series," in *The Handbook of Brain Theory and Neural Networks*, M. A. Arbib, Ed., MIT Press, 1995.

[27] A. Coates, B. Huval, T. Wang, D. J. Wu, and A. Y. Ng, "Deep learning with COTS HPC systems," in *Proceedings of the 30th International Conference on Machine Learning (ICML'13)*, vol. 28, pp. 1337–1345, JMLR: W&CP, Atlanta, Ga, USA, 2013.

[28] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15)*, pp. 1–9, Boston, Mass, USA, June 2015.

[29] R. Mottaghi, X. Chen, X. Liu et al., "The role of context for object detection and semantic segmentation in the wild," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014*, pp. 891–898, USA, June 2014.

[30] F. Xia, J. Wang, X. Kong, Z. Wang, J. Li, and C. Liu, "Exploring human mobility patterns in urban scenarios: a trajectory data perspective," *IEEE Communications Magazine*, vol. 56, no. 3, pp. 142–149, 2018.

Hindawi
Complexity
Volume 2019, Article ID 9301420, 14 pages
https://doi.org/10.1155/2019/9301420

Research Article

Promoting Geospatial Service from Information to Knowledge with Spatiotemporal Semantics

Jing Geng,[1,2] **Shuliang Wang**,[1,2] **Wenxia Gan**,[3] **Hanning Yuan**,[1] **Zeqiang Chen**,[4] **Ziqiang Yuan**,[1] **and Tianru Dai**[1]

[1] *School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China*
[2] *Academy of e-Government, Beijing Institute of Technology, Beijing 100081, China*
[3] *School of Civil Engineering and Architecture, Wuhan Institute of Technology, Wuhan 430074, China*
[4] *State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing (LIESMARS), Wuhan University, Wuhan 430079, China*

Correspondence should be addressed to Shuliang Wang; slwang2005@gmail.com, Wenxia Gan; gan.150@osu.edu, and Hanning Yuan; yhn6@bit.edu.cn

Received 9 November 2018; Accepted 8 January 2019; Published 21 January 2019

Guest Editor: Ke Deng

With the development of geoscience, users are eager to obtain preferred service from geospatial information intelligently and automatically. However, the information grows rapidly while the service gets more complicated, which makes it difficult to find out the targeted information for an exact service in geospatial issues. In this paper, a novel method is proposed to promote the geospatial service from information to knowledge with spatiotemporal semantics. Both prompted and professional knowledge are further refined to be published as a service. In terms of an exact task, numerous related services are recombined to a service chain under user requirement. Finally, the proposed method is applied to monitor the environment on the Air Quality Index (AQI) and soil moisture (SM) in the Sensor Web service platform, the results of which indicate geospatial knowledge service (GKS) is more efficient to support spatial decision-making.

1. Introduction

Nowadays, the geoscience issues benefit from the services of geospatial information. Users query the registered services on geospatial information, and the services are composited to form a service chain with the intervention from the users under a given task. But it is hard to meet users' requirements when it comes to accuracy, relevance, and time efficiency of the information. It is difficult to find the most relevant geospatial knowledge from the huge amounts of geospatial information. Lacking semantic attributes, the formalization requirement of geospatial information web service is also strict. In addition, it highly depends on users' manual intervention. Compared to information services, knowledge services pay more attention to the automation and intelligence of services with robust stability and fault tolerance. It is believed that knowledge service can make up for the lack of information services [1, 2]. In geographic information science (GIS), it is inevitable to develop from information services to knowledge services.

Normally, we need to acquire knowledge and get knowledge representation at first and then publish them as a service. At present, knowledge acquisition mainly includes interviews, simulation, oral records, multidimensional measurement from professional experts, engineering data handbooks, and modeling process. Acquisition emphasizes that experts can convert their knowledge or experience to an array of geospatial knowledge [3]. The development of machine learning additionally makes it possible to acquire knowledge with professional tools [4]. Such methods are time-consuming and face difficulty in meeting the requirement of knowledge acquisition, while spatial data mining might precisely solve the problem. The knowledge representation method converts the abstract representation into a linguistic one. Lack of formalizing languages results in that the knowledge exists in specific programs only and the knowledge cannot

be reused. Common logic programming languages [5–7] and new expression languages [8] are used for knowledge representation and knowledge reasoning. However, these programming languages have strict requirements on the platform and limit cross-platform knowledge applications. Semantic web can satisfy users' needs for geospatial knowledge and cross-platform application reuse [9–12]. Semantic web is a self-description independent meta markup language that is supported by its special formal description ability. It is conducive to analyzing and regularizing of geospatial knowledge.

Summarizing the state of the art on geospatial services, Gong et al. [13] presented geospatial knowledge service (GKS) that is user-oriented. GKS mainly serves a user with the corresponding solution from the geospatial knowledge. The user no longer needs to determine the required resources by themselves in complex and intricate information. Based on semantic relations, GKS not only automatically realizes service query, service reasoning, and service composition, but also intelligently provides users with the targeted information. To achieve GKS, all information and operations should be described and encapsulated as a service and published on the web. When there is a task, geospatial knowledge service will query, select, combine, and execute to respond to user's requirements [14–17]. Currently, the common description languages for semantic web service are OWL-S (Ontology Web Language-Semantic), WSMO (Web Services Modeling Ontology), SAWSDL (Semantic Annotations for Web Services Description Language), and XML (Extensible Markup Language) schema [13, 18–20].

A task requires not only service query and selection, but also service composition. The service query makes use of the spatiotemporal information to achieve efficient queries and inferences. Then the optional services are found out based on web server [21]. The service composition method is divided into process-driven service composition and semantic movement. The former is based on process modeling in the recombined services that aims at control flow and data flow, while the latter relies on semantics which focuses on semantics description and reasoning [22]. Amounts of common criteria are laid down to regulate the combination of services [23, 24], taking the basic criteria, for example, WSFL (Web Services Flow Language), WS-BPEL (Web Services-Business Process Execution Language), and BPMS (business process management system) [24–26].

However, the development of geospatial knowledge service is generally limited. There are still many fundamental problems in geospatial knowledge, e.g., the reasoning of knowledge-based services [27, 28], intelligent service compositions [29, 30], and the high-efficiency service composition mechanism [31, 32]. The intensive methods for geospatial knowledge service which are modeling, reasoning, and managing are compulsory [14, 33].

Therefore, a novel method is proposed to promote the geospatial service from information to knowledge with spatiotemporal semantics. The rest of this paper is organized as follows: the fundamental principles of the proposed method are described in Section 2. The application in environmental monitoring on the AQI and SM by using the proposed method is explained in Section 3. The conclusion and future plan are included in Section 4.

2. Fundamental Principles

The proposed method firstly promotes the geospatial service from information to knowledge with spatiotemporal semantics. Then, both prompted and professional knowledge are further refined to be published as a service. Given an exact task, several related services are recombined to a service chain under user requirement.

2.1. Geospatial Knowledge Service. Geospatial knowledge service (GKS) can rapidly query and access geospatial information by enhancing the semantic information on spatial distribution and serial trend. Its semantic reasoning with spatiotemporal constraint assists the service selection and composition, and both semantic information and prior knowledge are added for bettering services. To achieve an intelligent GKS, many services are composited to make a service chain for providing more complicated service automatically. GKS contains the concept model, technology, mode, infrastructure, and application on service (Figure 1).

The conceptual model defines the denotation and connotation of GKS. It is an abstract description of geospatial issues without any specific geospatial information processing program. Geospatial knowledge is stored in a specific resource management database which is accessible to applications that utilize metadata services, knowledge content services, and knowledge processing services separately or together, to provide geospatial knowledge to users.

The technology is that, to utilize GKS, users need to understand the formal representation, automatic identification, and automatic composition of geospatial knowledge. The criteria of ontology referring from semantic web are used to formalize the geospatial knowledge representation. The related technology and GIS are utilized to share and integrate, analyze, and process geospatial knowledge.

The mode refers to the interaction modes between the GKS platform and the application client, and it has three types: (1) centralized mode, (2) distributed mode, and (3) mobile agent mode [34–36]. The centralized mode concentrates on a single server or a small local area network, and the centralized server provides all services. The distributed mode distributes knowledge services on the wide area network, which enables the interactions between the servers and clients, and realizes the function of network management collaboratively. The mobile agent allows the user to provide relevant communication and transactions on the agent platform.

The infrastructure refers to the geospatial service web (GSW) which is a virtual geospatial infrastructure based on the internet that integrates various geospatial-related resources (sensors, data, processing programs, information, knowledge, computing resources, network resources, and storage resources). GKS can manage data, extract information, and obtain knowledge in the geospatial community domain [37]. GSW consists of five parts: geospatial

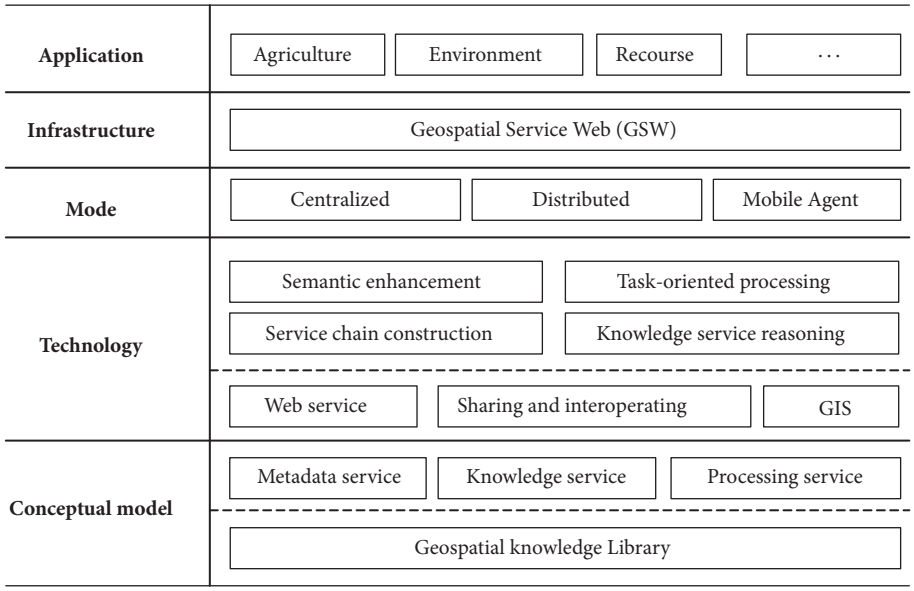


Figure 1: Geospatial knowledge service (GKS).

information resources, geospatial service, geospatial applications, GSW interoperability, and security standards. GSW offers comprehensive services about resources, information, applications, etc., where the functions are implemented by web services and communicated through the standardized protocols of the internet. Under the GSW, all resources are packaged as a service and published on the web. Users can choose corresponding services according to their own needs without understanding the resources, which can easily manage and analyze services.

The application of GKS covers various industry fields which can provide powerful online services, scalable geospatial analysis, and other auxiliary support. And it can be used on mobile phones, computers, personnel, etc. GKS can provide more information than the traditional information system and management decision suggestions. With GKS, users can realize the necessary procedures at the same time, such as information acquisition, management, processing, and analysis.

2.2. Promoting the Geospatial Service from Information to Knowledge with Spatiotemporal Semantics. The semantic enhancement of service is on the basis of GKS, with which the semantic function will be added to the original service. For instance, OWL-S is used to utilize the service configuration file, service model, and services infrastructure to realize the semantic web service. As a result, the generated logical description can be recognized by programs which make it possible to achieve intelligent services.

The logical description of OWL-S includes the service profile, service model, and service binding [38, 39] in Figure 2: (1) the service profile explains the content of the web service, including its usage and function, (2) the service model provides the service execution logic, including the executing sequence, and (3) the service binding rules introduce the rules for invoking the web service, which is

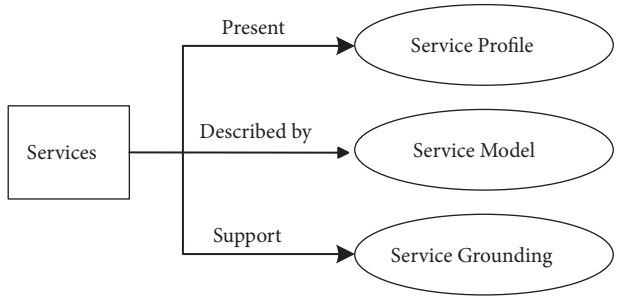


Figure 2: Semantic-enhanced service.

mainly the specific binding information, such as message format, communication protocol, and URL.

2.3. Publishing Prompted and Prior Knowledge as a Service. We publish promoted and prior knowledge as a service. The black box approach was implemented to facilitate the publishing of knowledge to web services. The black box approach [40] can be used without understanding how its inner algorithm works; the user only needs to know the input and output characteristics.

The "inside" of the black box is critical for generating web services with their names, inputs, outputs, and executable programs. The black box is accessed by a visual user interface, and the outline for mapping the relationship between a model and a web service is shown in Figure 3. The input, output, and execution code of a model are mapped to the input, output, and execution code of a web service, respectively. The input and output parameters are consistent with the executable program inputs and outputs of the black box.

For interface implementation, a black box can be deployed in many different forms (e.g., executable file (EXE), script, dynamic link library (DLL), or other written program languages). These form-executable programs are essential to

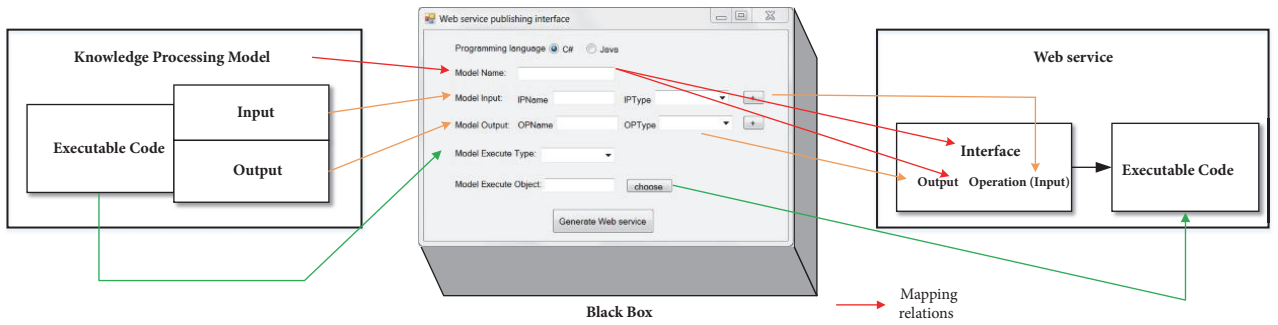


Figure 3: Mapping a model to a web service.

Table 1: Table of a service chain.

Entry	Type	Notion
ID	long	Unique identification
Service _id	varchar	Related service
relationship_id	varchar	Related relationship
MD_id	varchar	Related metadata
UNIQUE(Service_id, relationship_id, MD_id)		Unique constraint key
PRIMARY KEY (ID)		Primary key

running a local black box. The web service development library could support the development of web services. And the open source project of Java, such as Axis2, can be used to develop and deploy web services.

2.4. Creating a Service Chain under Task Requirement. With the explosive growth of knowledge, a single service cannot meet the needs of users. It is necessary to aggregate multiple services and form a service chain to provide a solution. This section presents the expression, storage, construction, and management of the service chain.

2.4.1. Task-Oriented Service Chain Construction. A service chain is an aggregation of multiple services based on logic and it can be divided into two types: abstract service chain and specific service chain. An abstract service chain completely describes the concept of the service chain and its logical relationship. A specific service chain integrates each service in the abstract service chain with a concrete service and then uses the service chain engine to execute.

The database-based storage and construction of the service chain are developed in light of the database that facilitates internet access. To store both the services and their relationships, the service chain storage tables are designed to construct a service chain in the database. Their structures are shown in Tables 1–3.

As many of their entries are complex and connect with other tables, the definition $*$_id is used to define the table's entry, which is applied in other tables.

The construction method for the abstract service chain and specific service chain is developed, respectively. The builder needs to know the processes involved and the relationships between these processes precisely. Based on these processes and their relationships, the builder constructs the service chain and registers it to the library by utilizing the service chain expression method and service chain storage method.

The following three steps are carried out to construct the abstract service chain: (1) Analyze the content of each process and clarify the links between them. (2) Create the abstract element for each process. Once the service chain builder has selected a series of processes, it is necessary to decompose the series into single processes abstractly. (3) Establish the relationship between the processes. After abstracting the processes, the tuple representation of the service chain is entirely expressed by defining the relationships between the processes.

A visual method is employed to construct an abstract service chain, which utilizes icons and human-computer interaction to complete the construction simply and briefly. In the service chain constructing panel, the visual drag module and the relationship building module are predeveloped, which are shown as red boxes and arrows in Figure 4.

There are three steps to construct the abstract service chain: (1) for any given task and its corresponding processes, drag the red boxes representing the required processes to the panel; (2) submit the required information by operating the red box to clarify the function, name, and temporal data of the processes; and (3) create the relationship among the processes by dragging the relationship icons and providing the related information.

The specific service chain can be built based on the abstract chain, or by combining the specific services directly. So there are two service chain construction methods. The first is to construct a specific service chain with the abstract service chain. Its core is to bind the concrete service to the abstract service chain and integrate the information of the specific service to the process and the relationship. When an abstract service chain is established, the icons for the abstract service processes and relationships appear in the opened operation panel, as shown in Figure 5. The detailed steps of this method are as follows: (1) Initialize the services and set the parameters of the abstract services. This operation is realized in the dialog box that emerges after right-clicking the icon. (2) Clarify the relationship between the input

Table 2: Table of a service.

Entry	Type	Notion
ID	long	Unique identification
types_id	varchar	Related types
inport_id	varchar	Related import
message_id	varchar	Related message
porttype_id	varchar	Related portType
operation_id	varchar	Related operation
binding_id	varchar	Related binding
Service _id	varchar	Related
MD_id	varchar	Related metadata
UNIQUE(types_id, input_id, message_id, porttype_id, operation_id, binding_id, Service _id, MD_id)		Unique constraint key
PRIMARY KEY (ID)		Primary key

Table 3: Table of a service relationship.

Entry	Type	Notion
ID	long	Unique identification
Service _id	varchar	Related service
Service _id	varchar	Related service
MD_id	varchar	Related metadata
r	string	Relationship between services (optional pattern, circulation pattern, sequential pattern)
UNIQUE(Service_id, Service_id)		Unique constraint key
PRIMARY KEY (ID)		Primary key

and output and the concrete services to form a concrete executable logical sequence.

The second is to directly construct the service chain with specific services. If all the services are known, there is no existing abstract service chain to combine them. The service chain can be constructed directly by combining these known services (Figure 6). The detailed steps of this method are as follows: (1) Drag the service icon to initiate the specific service, and then right-click the icon and enter the WSDL of the service. The program will automatically analyze the WSDL and generate the related information about the specific services. (2) Establish a service node to analyze WSDL automatically. Clarify the operation and the input-output relationship of the specific services to bind the relationships between specific services.

After the service chain is established, the dynamic updating of the associated tables makes it simple to add, modify, or maintain the service chain. The visual management method presents the simple service chain diagram to manage the service chain. The dynamic management of the service chain consists of two major steps: (1) extracting the data contained in the service chain and its related relationship and, (2) based on the specific conditions, dynamically updating all the tables and information involved in the service chain.

A GKS can contain a single service or multiple services. Under both conditions, the proper service must be entered orderly to achieve a given objective. Knowledge reasoning methods may help achieve the reuse of geospatial knowledge. Introducing the spatiotemporal characteristics of geospatial knowledge, the semantic spatiotemporal reasoning rules are created by extending Jena's reasoning rule with spatiotemporal constraints and benefit the service selection accuracy improvement. Meanwhile, the service composition reasoning rules are proposed based on the existing reasoning rules.

2.4.2. Spatiotemporal Semantic Reasoning Rules for Service Selection. In this study, the semantic spatiotemporal reasoning rule based on OWL-S is used to realize the discovery and binding of GKS. Like the query model shown in Figure 7, Profile, Service Name, Service Parameter, Layer, Feature, Coverage, Observation, BBox, and Time are all classes which have both data and objects attributes. The query relies on keywords such as the type of service, time and space conditions, categories, and titles, which correspond to the Service Name, BBox, Time, CategoryName, and Title in the model. Those keywords can be used separately or together.

Supported by Jena's reasoning machine, we defined the following query rules based on OWL-S.

Rule 1. Relationship between "CategoryName" and "Profile."

Rule 2. Relationship between "Thing" and "Profile."

Rule 3. Relationship from "Layer," "Feature," "Coverage," and "Observation" to "BBox."

Rule 4. Relationship from "Layer," "Feature," "Coverage," and "Observation" to "Time."

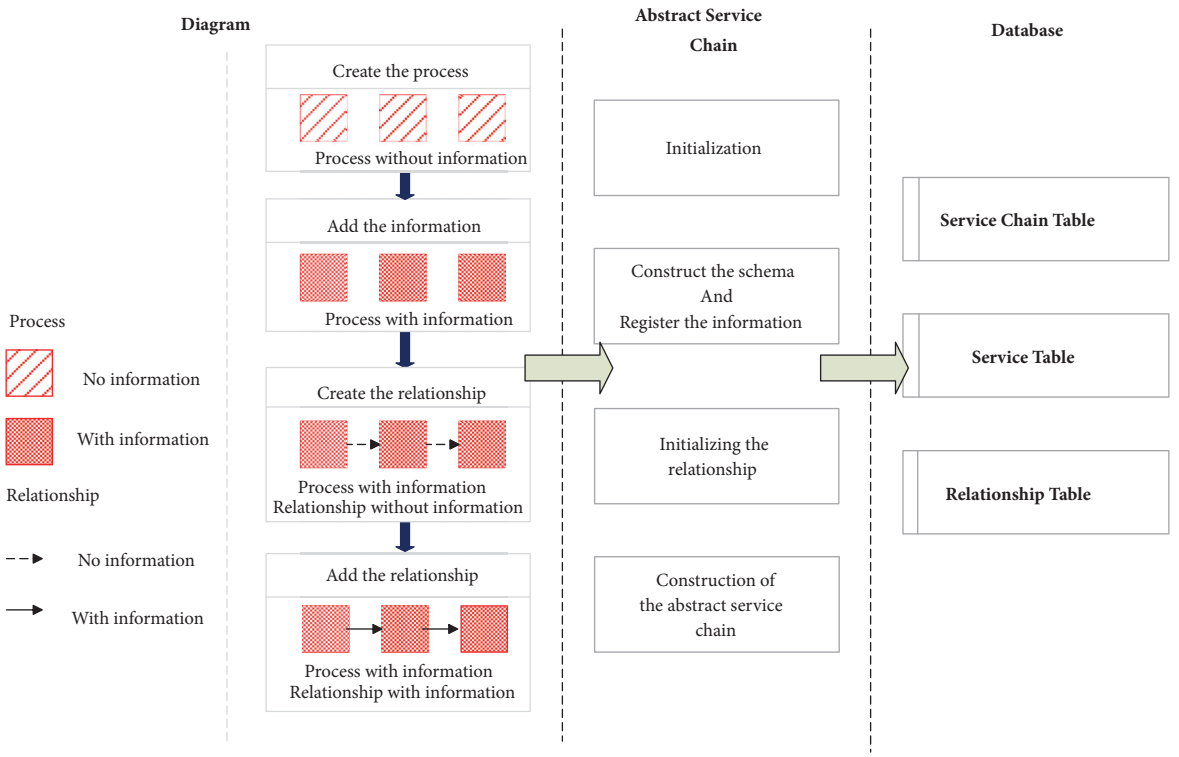


Figure 4: Construction of the abstract service chain.

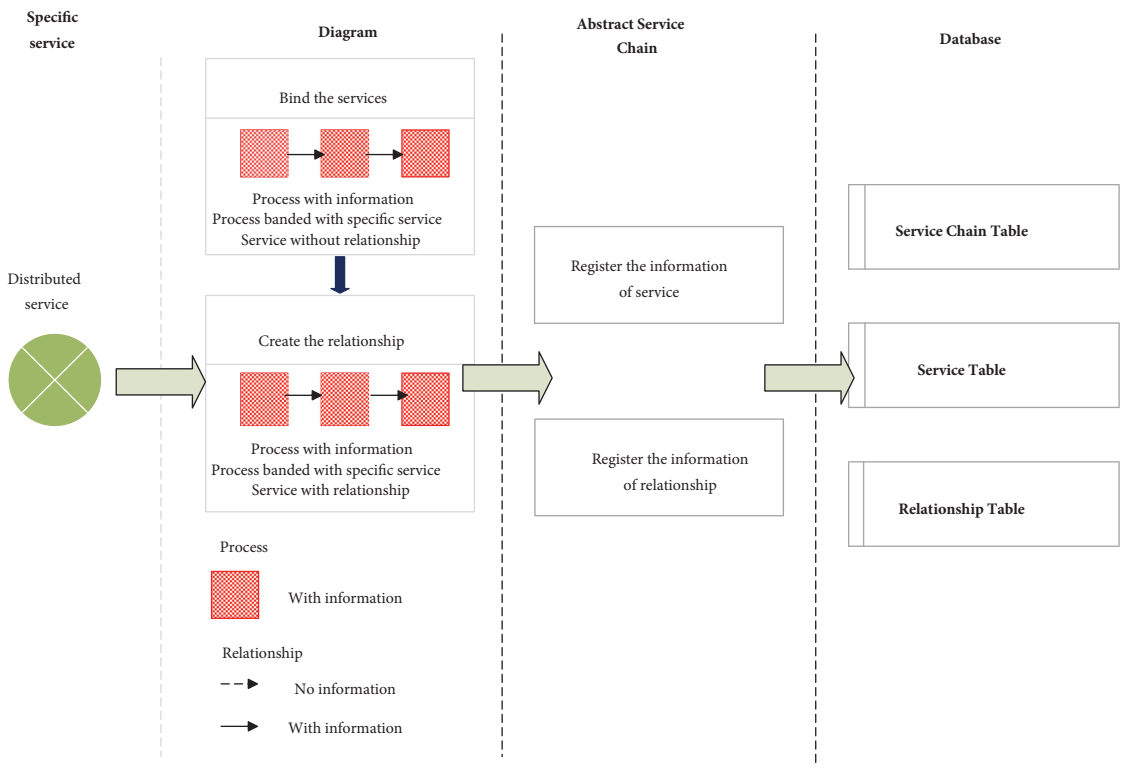


Figure 5: Construction of a specific service chain based on the abstract service chain.

Table 4: Different rules for different query parameters.

Query parameter	Rule
Service type	Rule 1
Spatial range	Rule 3, Rule 2
Temporal range	Rule 4, Rule 2
Title	Rule 5, Rule 2
Service type, spatial range	Rule 3, Rule 6, Rule 1
Service type, temporal range	Rule 4, Rule 6, Rule 1
Service type, title	Rule 6, Rule 2
Spatial range, temporal range	Rule 3, Rule 4, Rule 2
Spatial range, title	Rule 3, Rule 2
Temporal range, title	Rule 4, Rule 2
Service type, spatial range, temporal range	Rule 3, Rule 4, Rule 6, Rule 1
Service type, spatial range, title	Rule 3, Rule 6, Rule 1
Service type, temporal range, title	Rule 4, Rule 6, Rule 1
Spatial range, temporal range, title	Rule 3, Rule 4, Rule 2
Service type, spatial range, temporal range, title	Rule 3, Rule 4, Rule 6, Rule 1

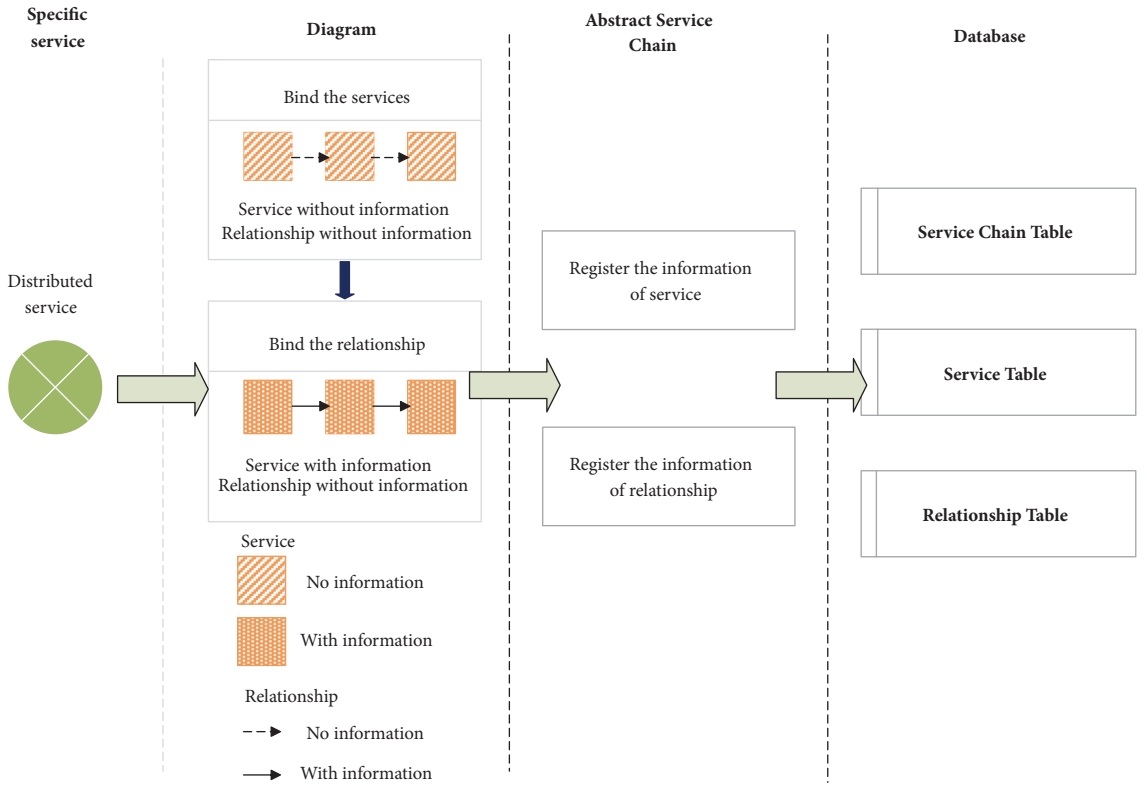


Figure 6: Construction of a service chain based on specific services.

Rule 5. Relationship from "Layer," "Feature," "Coverage," and "Observation" to "Title."

Rule 6. Relationship between "CategoryName" and "Thing."

The query parameters (the service type, spatial range, temporal range, and title or their compounds) and its corresponding rules are illustrated in Table 4. For example, when query parameters are spatial range, temporal range, and title, the instances of "CategoryName" and "Thing" are found with Rule 6 and then the Time instances as specified by the data property of the instance of "Thing." Then, the Time instance of "Title" with the instances of "Thing" and Rule 4 are found. At last, the BBox instance of "Title" with the instances of "Thing" and Rule 3 are found, as well as the instance of "Profile" with Rule 1, and thus the URL of the web service was found.

2.4.3. Semantic Spatiotemporal Reasoning Rules for Service Composition. The service composition method selects the

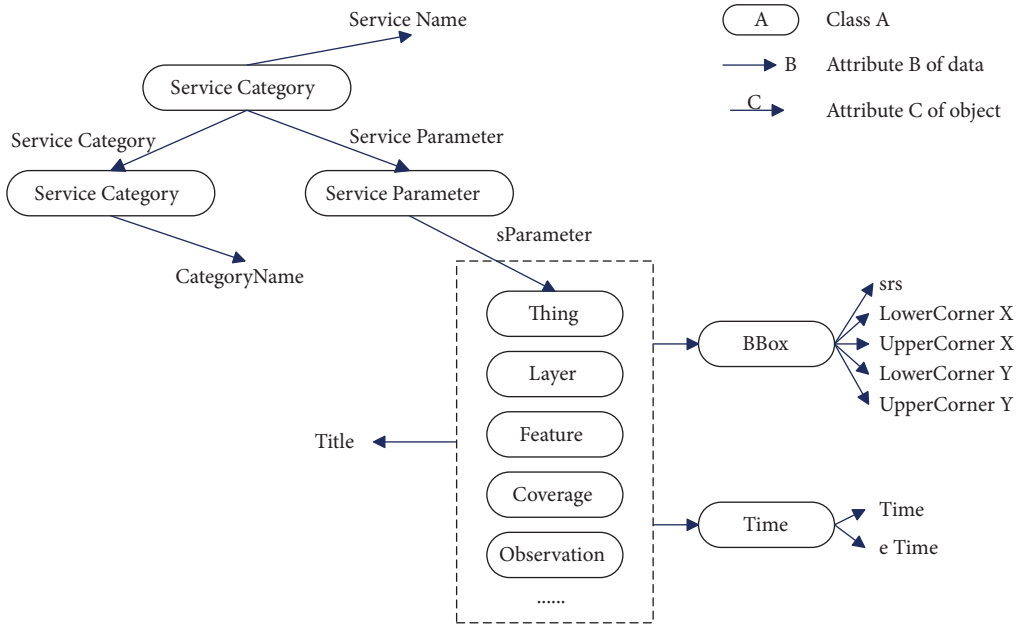


Figure 7: The query model based on OWL-S reasoning.

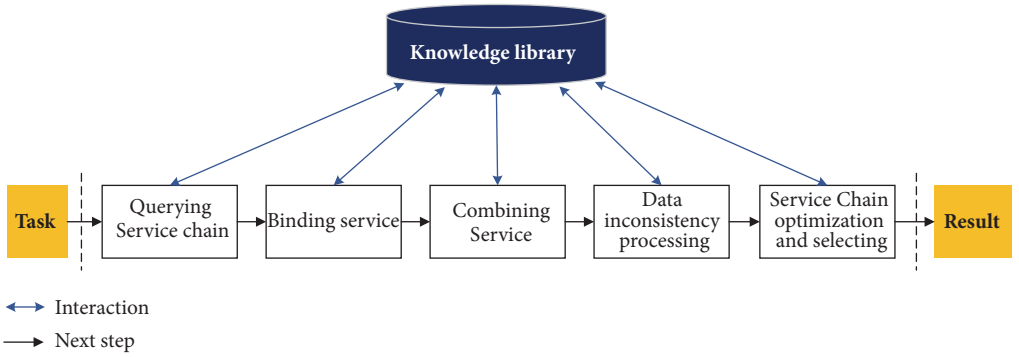


Figure 8: Workflow of semantic reasoning of GKS composition.

suitable services from a number of service sets and combines these services automatically in accordance with their inherent rules. The artificial intelligence and the workflow generation approach are employed to realize intelligent service compositions. To obtain the results of a specific task, there are 6 steps in Figure 8: (1) Query the service chain or single service from the knowledge library, according to the demands of the task. (2) Bind the abstract service with a specific service if necessary. (3) Combine the services and convert instances of the services to form a combined executable service. (4) Process data inconsistency. Adjust the compositing processing of web services to deal with the data inconsistencies among different services. (5) Optimize and select service chain. Find the optimal composition of services from the multiple possible permutations for services composition. (6) Output the results.

The reasoning method used in this workflow includes the service chain selecting inference, service binding inference, service compositing inference, inconsistency eliminating inference, and service optimal select inference.

(1) The main goal of the service chain selecting reasoning method is to match the task, queries, and the usage of the service or service chain, including semantic matching and matching based on grammatical keywords

(2) The binding service reasoning method combines the service chain with the service instances and the abstract service chain with specific services automatically, instead of relying on human interaction. The reasoning method is similar to the service chain selecting method, mainly through the grammatical keyword matching between the name of the service and the instance. When the process is successful, the chosen services are banded directly; but if it fails, it is necessary to check for another matching service in the semantic knowledge library

(3) Service composition reasoning transfers the semantic chains to an executable workflow chain. Since the service is operated through an interface, three basic composition relationships between two interfaces are proposed. The first one: if the output of a specific operation in an interface is part or all of the input of another interface, indicating

Table 5: Data inconsistency table.

data1	data2	Data inconsistency
Vector data	Raster data	Data type
Raster data	Vector data	Data type
Vector data	Vector data	Coordinate system
Raster data	Raster data	Coordinate system
		Resolution
		Data format

these two interfaces are associated. The composition of these two interfaces have to obey the specific order. The second one: if the input of an interface is the outputs of both interfaces, the two interfaces have to work together to form a collaborative relationship. The third one: based on these composite relationships, multiple web services are integrated for each task

(4) Data inconsistency processing forms the data flow of the service chain. Two commonly used data types in GIS and remote sensing (RS) are vector and raster data. Data inconsistencies of GIS and RS are shown in Table 5.

A transformation service is employed to overcome data inconsistency, which includes data type transformation, coordinate system transformation, data format transformation, and resolution transformation. In order to solve this problem automatically and achieve the data transformation, a metadata model is proposed, which consists of data type (DT), satellite/sensor type (ST), coordinate system (CS), resolution (RE), and data format (DF)

(5) The service optimal select reasoning method intends to improve efficiency by choosing and optimizing the integration processes of the web services needed. For example, if a task has many web service composition schemes, the objective is to determine the scheme with the minimal cost. Zeng et al. came up with five generic quality criteria for elementary services including execution price, execution duration, reputation, reliability, and availability [41]. They also selected a global optimal execution scheme. According to the features of GKS, the global optimal execution time is the primary consideration. The final service chain schemes are the outputs of the workflow. The process of reasoning rule is as follows: (1) Query the relevant abstract service chains through the semantic query. (2) Bind the corresponding services to the service chains. This process might link to other services. Thus, a variety of possible specific service chains might be formed. (3) Calculate the cost of different schemes, and then choose a specific service chain to perform the task

3. Application in Environmental Monitoring on AQI and SM

As serious environmental problems emerge frequently, this has become harmful to human health and caused amounts of economic loss. Government officials and citizens have been paying more attention to air quality than ever before. The availability of environmental monitoring data could offer useful information and rapidly respond to environmental events. And the comprehensive knowledge is vital for officers to propose a proper and efficient management plan and schema. This section focuses on the application of GKS in environmental monitoring.

Air pollution is a well-known environmental problem in the whole world, where the Air Quality Index (AQI) and Air Pollution Index (API) are two common indexes to reflect the air quality. Soil moisture (SM) is an important environmental indicator for studying climate change, indicating the degree of agricultural drought, and guiding agricultural irrigation. Monitoring the SM conditions in the whole experiments will assist the decision-making of the drought degree in the area. Air quality and soil moisture monitoring are explored in Wuhan, China.

The governmental agency named Wuhan Environmental Monitoring Centre established some environmental monitoring stations and deployed many sensors in Wuhan to monitor SO_2, NO_2, PM10, CO, O_3, and PM2.5 pollutants. As to SM, an automatic observation station with more than 20 soil moisture sensors was deployed in horizontal planes in a 20 m × 40 m experimental area (center location at 114°31′ 35. 61″E 30°28′ 12. 98″N) in Baoxie town, Wuhan.

3.1. Publishing Prior Knowledge as a Service. The computing of the AQI and SM is published as a service in the platform, which is implementation with Sensor Web technologies by the Sensor Web Group of Wuhan University, China [42, 43].

(1) Air Quality Index (AQI). The United States Environmental Protection Agency has released a guideline for standardizing the AQI, individual AQI calculation methods, and category descriptors. The calculation method was illustrated as follows:

$$\mathrm{AQI} = \mathrm{MAX}\left(I_1, I_2, \cdots, I_n\right) \tag{1}$$

$$I_P = \frac{I_{Hi} - I_{Lo}}{BP_{HI} - BP_{LO}}\left(C_P - BP_{LO}\right) + I_{LO}, \tag{2}$$

where I_P is the exponent of the value of pollutants, C_P is the floor of the value of pollutants, BP_{HI} and BP_{LO} are the upper breakthrough value and the lower breakthrough value of C_P, respectively, and I_{HI} and I_{LO} are the corresponding AQIs of BP_{HI} and BP_{LO}.

(2) Soil Moisture (SM). The deployed soil moisture sensors only monitor discrete points in the area, and their number is limited. Inverse distance weighted (IDW) interpolation was adopted to interpolate values at unobserved points in the experiments. The IDW interpolation is under the assumption that the attribute value of a point is weighted related to the values of its neighborhood, and the weights are inversely related to the distances between the predicted location and the sampled locations.

Assume the sampled point is represented as $P_n = \{x_n, y_n, v_n\}$, where x_n, y_n are its location information and v_n is its attribute value. For the required interpolation point P = $\{x, y, v\}$, if x and y are known, v can be calculated using the equations below.

$$v = \sum_{i=1}^{n} w_i v_j, \tag{3}$$

where v is the SM value, v_j is the SM value of the neighbor point j, n is the number of the neighbor points, and w_i is the weight calculated by using the following equation:

$$w_i = \frac{1/d_i}{\sum_{i=1}^{n} (1/d_i)}, \tag{4}$$

where the distance d_i is calculated by

$$\mathrm{d}_i = \sqrt{(x - x_i)^2 + (y - y_i)^2}. \tag{5}$$

According to the method introduced in Section 2.2, the calculation of AQI and the interpolation of SM were published as a web service. The sensors were registered into Sensor Observation Service (SOS) [44], and then the data from the sensors were inserted into SOS. The input of the AQI calculation service is the observed concentration of various individual pollutants, and its data format is observations and measurements (O&M). The input of the SM interpolation service is the value of the observed point and the regional grid image, in which the observation data format is O&M, and the output result is raster data with soil moisture data in each grid.

3.2. Semantic-Enhanced Knowledge Service. As introduced in Section 2.2, the service configuration file, service model, and services infrastructure are used to realize the semantic enhancement of web services. Supported by OWL-S, semantic annotations (e.g., title, parameters, temporal range, and spatial range) are added to all of the Sensor Observation Service, the AQI calculation service, the soil moisture interpolation service, etc.

3.3. Task-Oriented Services Recombination in a Service Chain. GKSs and service chain are supposed to register on the content library and knowledge library, respectively.

The AQI service chain includes standardized AQI and IAQI calculation service. According to the service chain regularization method introduced in Section 2.4, the AQI calculation can be described as SC={ID, S, R, MD}, where SC is the AQI monitoring service chain, ID is a globally unique identification assigned by the program automatically, and S is the set of services, including the observation service of SO_2, NO_2, PM10, CO, O_3, and PM2.5. The relationship set R={C, S} consists of the collaborative relationship among the observation services and the sequential relationship between the observation service and the AQI calculation service.

The SM monitoring service chain contains interpolation and cartography service; it can be expressed as SC= {ID, S, R, MD}, where SC is the SM monitoring service chain, R={C} is the sequential relationship between the interpolation and the cartography service, and MD is the corresponding metadata.

3.4. Results and Analysis. The air quality monitoring experiment started from 2014-09-08 14:00 to 2014-09-10 15:00; and for soil moisture, it was from 2014-07-05 to 2014-07-07. The data used in this experiment was provided by the Wuhan Environmental Monitoring Centre.

Firstly, the AQI and SM monitoring service chain were queried from the knowledge library. The queried service chains are the combination of abstract and specific services. Queried by the keywords "the air quality of Wuhan city from 2014-09-08 14:00 to 2014-09-10 15:00" and "the soil moisture of Baoxie in Wuhan city from 2014-07-05 to 2014-07-07," following the reasoning method introduced in Section 2.4, the air quality monitoring service of Wuhan and the soil moisture service of Baoxie were achieved, as shown in Figures 9-10 (created from MapWorld: http://www.tianditu.cn/).

As can be seen in Figures 9-10, the refresh rate is 10 seconds. Real time environmental data are measured by three air quality indicators including AQI 1, PM2.5, and PM10. Information of sensors contains name, expected application, work status, keywords, loading sensors, organization, schema, detailed information, and so on. According to the 24 h AQI line graph, the value of AQI peaked at approximately 190 between 15 pm to 17 pm since this period of time is the normal commuting rush hour in China. And the value of the AQI reached the lowest point around 22 pm in Wuhan, China. Compared with the 48 h AQI line graph, the value of the AQI shows the same trend. And different indicators are used to measure SM. The user can choose different time periods to visualize the temporal observation of SM.

The specific observation service and processing service were banded together to achieve the optimal service chain. And the air monitoring and SM monitoring results were shown in Figure 11. The user has to choose the temporal range and quarrying time at first, and then the corresponding status of the map can be seen on the monitor. The information sensor includes Num, Sensor, Platform, Value, and Time. And each piece of sensor data can be visualized as a scatter graph.

4. Conclusion

In this paper, a novel method is proposed to promote the geospatial service from information to knowledge with spatiotemporal semantics. Both prompted and professional knowledge are further refined to be published as a service. In terms of an exact task, several related services are recombined to a service chain by using spatiotemporal enhanced reasoning under user requirement. An air environmental and soil moisture monitoring application was equipped in Wuhan, China, and proved the flexibility of GKS and successfully reached satisfaction. However, due to the complexity and comprehensive nature of GKS, the related theory should be improved, for example, the method for constructing GKS domain ontology. The reasoning method proposed in this paper still needs the prior knowledge and is not fully automated and intelligent. Thus, importing artificial intelligence technology to GKS is our next step.

Data Availability

The AQI monitoring and SM data used to support this study were supplied by Wuhan Environmental Monitoring Centre

Figure 9: Air quality monitoring.

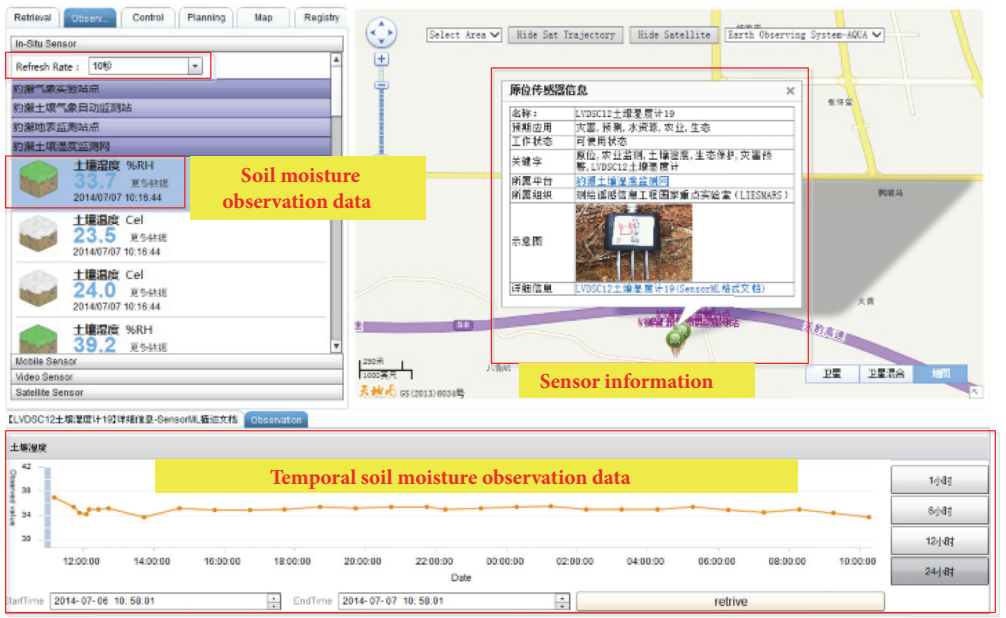


Figure 10: Observation of soil moisture.

Figure 11: Air monitoring and soil moisture monitoring results.

under license and they are not available. Requests for access to these data should be done through contacting Wuhan Environmental Monitoring Centre, Tel. + 86 27 85805108.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (2016YFB0502600), the National Natural Science Fund of China (61472039, 41701415), and the Open Fund of Key Laboratory for National Geographic Census and Monitoring, National Administration of Surveying, Mapping and Geoformation (2017NGCMZD03).

References

[1] D. Li, S. Wang, and D. Li, *Spatial Data mining Theory and Application*, Springer Publishing Company, Incorporated, 2016.

[2] G. Jianya, "Review of the progress in contemporary GIS," *Geomatics & Spatial Information Technology*, vol. 27, no. 1, pp. 5–11, 2004.

[3] C.-h. LiU, S.-d. Wang, and H.-t. Zhang, "Theoretical and technical issues on intelligent processing and analyzing models for temporal and spatial data," *Journal of Image & Graphics*, 2001.

[4] T. Shanxin, "Research and implementation on product developing system based on knowledge driven," *Computer Engineering & Applications*, vol. 8, no. 12, pp. 295–298, 2003.

[5] S. Zongyao and B. Fuling, "Object-oriented spatial knowledge representation and its application," *Journal of Remote Sensing*, 2004.

[6] U. Visser, H. Stuckenschmidt, G. Schuster, and T. Vögele, "Ontologies for geographic information processing," *Computers & Geosciences*, vol. 28, no. 1, pp. 103–117, 2002.

[7] A. Ma, "Formalization of geographical knowledge," *Science of Surveying and Mapping*, vol. 26, no. 4, pp. 8–12, 2001.

[8] P. Mancarella, A. Raffaetà, C. Renso, and F. Turini, "Integrating knowledge representation and reasoning in Geographical Information Systems," *International Journal of Geographical Information Science*, vol. 18, no. 4, pp. 417–446, 2004.

[9] S. Kona, A. Bansal, and G. Gupta, "Automatic composition of semantic web services," in *Proceedings of the 2007 IEEE International Conference on Web Services, ICWS 2007*, pp. 150–158, USA, July 2007.

[10] N. Shadbolt, W. Hall, and T. Berners-Lee, "The semantic web revisited," *IEEE Intelligent Systems*, vol. 21, no. 3, pp. 96–101, 2006.

[11] A. Maedche and S. Staab, "Ontology Learning for the Semantic Web," *IEEE Intelligent Systems*, vol. 16, no. 2, pp. 72–79, 2001.

[12] H. Knublauch, R. W. Fergerson, N. F. Noy, and M. A. Musen, "The Protégé owl plugin: an open development environment for semantic web applications," in *Proceedings of the International Semantic Web Conference*, Springer, 2004.

[13] J. Gong, J. Geng, and H. Wu, "Geospatial knowledge service: A review," *Wuhan Daxue Xuebao (Xinxi Kexue Ban)/Geomatics and Information Science of Wuhan University*, vol. 39, no. 8, pp. 883–890, 2014.

[14] S. K. Abujayyab, M. A. Ahamad, A. S. Yahya, and A. H. Saad, "A new framework for geospatial site selection using artificial neural networks as decision rules: a case study on landfill sites," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. II-2/W2, pp. 131–138, 2015.

[15] X. Tan, L. Di, M. Deng et al., "Agent-as-a-service-based geospatial service aggregation in the cloud," *Environmental Modelling & Software*, vol. 84, pp. 210–225, 2016.

[16] O. Chakraborty, J. Das, A. Dasgupta, P. Mitra, and S. K. Ghosh, "A geospatial service oriented framework for disaster risk zone identification," in *Proceedings of the in International Conference on Computational Science Its Applications*, 2016.

[17] J. Wagemann, O. Clements, R. Marco Figuera, A. P. Rossi, and S. Mantovani, "Geospatial web services pave new ways for server-based on-demand access and processing of Big Earth Data," *International Journal of Digital Earth*, vol. 11, no. 1, pp. 7–25, 2018.

[18] K. I. S. Harrington, C. T. Rueden, and K. W. Eliceiri, "FunImageJ: A Lisp framework for scientific image processing," *Bioinformatics*, vol. 34, no. 5, pp. 899-900, 2018.

[19] L. Chaohui, L. Rui, and W. Jingqi, "A dynamic representation method of considering semantic scales of attributes of spatio-temporal object," *International Journal of Geographical Information Science*, vol. 19, no. 9, pp. 1185–1194, 2017.

[20] A. Luo, Y. Wang, and J. Gong, "A semantic matching method for geospatial information service composition based on context," *Wuhan Daxue Xuebao (Xinxi Kexue Ban)/Geomatics and Information Science of Wuhan University*, vol. 36, no. 3, pp. 368–372, 2011.

[21] J. Das, A. Dasgupta, S. K. Ghosh, and R. Buyya, "A Geospatial Orchestration Framework on Cloud for Processing User Queries," in *Proceedings of the 5th IEEE International Conference on Cloud Computing in Emerging Markets, CCEM 2016*, pp. 1–8, India, October 2016.

[22] X. Lianxia, *Research on Geographic Infromation Data Exchange Technique Based on Web Services*, Suzhou University, 2008.

[23] G. Tian, J. Wang, K. He, C. Sun, and Y. Tian, "Integrating implicit feedbacks for time-aware web service recommendations," *Information Systems Frontiers*, vol. 19, no. 1, pp. 75–89, 2017.

[24] H. Labbaci, B. Medjahed, and Y. Aklouf, "Learning interactions from web service logs," in *Database and Expert Systems Applications*, vol. 10439 of *Lecture Notes in Computer Science*, pp. 275–289, Springer International Publishing, 2017.

[25] A. Ahrabian, S. Kolozali, S. Enshaeifar, C. Cheong-Took, and P. Barnaghi, "Data analysis as a web service: A case study using IoT sensor data," in *Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6000–6004, New Orleans, LA, USA, March 2017.

[26] A. De Renzis, M. Garriga, A. Flores, A. Cechich, C. Mateos, and A. Zunino, "Assessing readability of Web service interfaces," in *Proceedings of the 2016 XLII Latin American Computing Conference (CLEI)*, pp. 1–12, Valparaíso, Chile, October 2016.

[27] M. C. Martinez-Fernandez, C. A. Soosay, M. Bjorkli et al., "Are knowledge-intensive service activities enablers of innovation processes?-a study of Australian Software Firms," in *Proceedings of the in CINet Referred Conference Proceedings Conference*, 2004.

[28] J. Choi, B. Kim, H. Hahn et al., "Data mining-based variable assessment methodology for evaluating the contribution of knowledge services of a public research institute to business performance of firms," *Expert Systems with Applications*, vol. 84, pp. 37–48, 2017.

[29] P. Stelmach, A. Grzech, and K. Juszczyszyn, "A model for automated service composition system in SOA environment," in *Proceedings of the Doctoral Conference on Computing, Electrical and Industrial Systems*, Springer, 2011.

[30] Y. Sam, O. Boucelma, and M. Hacid, "Semantic Web Services Composition for the Mass Customization Paradigm," in *Proceedings of the 8th International Conference on Enterprise Information Systems(ICEIS 2006)*, Palo Alto, Calif, USA, July 2006.

[31] A. Bundy, "Incidence calculus: a mechanism for probabilistic reasoning," *Journal of Automated Reasoning*, vol. 1, no. 3, pp. 263–283, 1985.

[32] N. Chen, Z. Chen, C. Hu, and L. Di, "A capability matching and ontology reasoning method for high precision OGC web service discovery," *International Journal of Digital Earth*, vol. 4, no. 6, pp. 449–470, 2011.

[33] K. Karantzalos, D. Bliziotis, and A. Karmas, "A scalable geospatial web service for near real-time, high-resolution land cover mapping," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 10, pp. 4665–4674, 2015.

[34] L. Birong and X. Debao, "Network management based on intelligent mobile agent," *Minimicro Systems (Shenyang)*, vol. 22, no. 7, pp. 864–867, 2001.

[35] Y. Chen, Y. Wu, Z. Li, and Y. Zhou, "Web service composition based on mobile agents," *Journal of Southeast University (Natural Science Edition)*, vol. 38, no. A01, pp. 284–287, 2008.

[36] F. Spolidoro and N. Rodriguez, "Distributed environment for web-based network management," *Conference on Local Computer Networks*, pp. 41–48, 2001.

[37] J. Gong, H. Wu, W. Gao, P. Yue, and X. Zhu, "Geospatial service web," *Geospatial Technology for Earth Observation*, pp. 355–379, 2009.

[38] W. Xin, "Key technologies on dynamic integration of service systems in digital libraries," *Library*, vol. 2, pp. 50–53, 2005.

[39] W. Xin and Z. Xiaolin, "Realizing semantic web services description with OWL-S ontology," *New Technology of Library and Information Service*, vol. 21, no. 2, pp. 15–19, 2005.

[40] B. Beizer and J. Wiley, "Black box testing: techniques for functional testing of software and systems," *IEEE Software*, vol. 13, no. 5, p. 98, 1996.

[41] L.-Z. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng, "Quality driven web services composition," in *Proceedings of the 12th International Conference on World Wide Web (WWW '03)*, pp. 411–421, ACM, May 2003.

[42] "The Sensor Web Common Service Platform," 2018, http://gsw.whu.edu.cn:9002/SensorWebProEng/.

[43] N. Chen, X. Yang, and X. Wang, "Design and implementation of geospatial sensor web information public service platform,"

International Journal of Geographical Information Science, vol. 15, no. 6, p. 887, 2013.

[44] A. Bröring, C. Stasch, and J. Echterhoff, "OGC® Sensor Observation Service Interface Standard," https://portal.opengeospatial.org/files/?artifact_id=47599, 2012.

Hindawi
Complexity
Volume 2019, Article ID 7643905, 14 pages
https://doi.org/10.1155/2019/7643905

Research Article

A Destination Prediction Network Based on Spatiotemporal Data for Bike-Sharing

Jian Jiang[1], **Fei Lin**[1], **Jin Fan**[1], **Hang Lv**[1] **and Jia Wu**[2]

[1]*School of Computer Science and Technology, Hangzhou Dianzi University, 310018, Hangzhou, China*
[2]*Department of Computing, Macquarie University, Sydney, Australia*

Correspondence should be addressed to Fei Lin; linfei@hdu.edu.cn

Received 2 July 2018; Revised 27 September 2018; Accepted 30 October 2018; Published 1 January 2019

Guest Editor: Jianxin Li

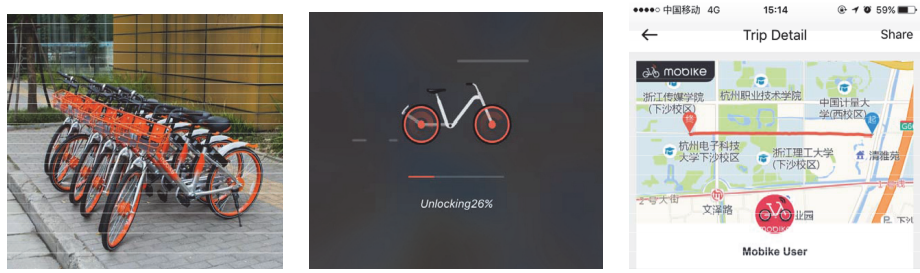
Bike-sharing is a new low-carbon and environment-friendly mode of public transport based on the "sharing economy". Since 2017, the bike-sharing market has boomed in China's major cities. Bikes equipped with GPS transmitters are docked along sidewalks that can be easily accessed through smartphone apps. However, this new form of transport has also led to problems, such as illegal parking, vandalism, and theft, each of which presents a major administrative challenge. Further, imbalances in user demand and bike availability need to be overcome to ensure a convenient, flexible service for customers. Hence, predicting a cyclist's destination could be of great importance to shared-bike operators. In this paper, we propose an innovative deep learning model to predict the most probable destination for each user. The model, called destination prediction network based on spatiotemporal data (DPNst), comprises three steps. First, the data is preprocessed and a pool of likely candidate destinations is generated based on frequent item mining. This candidate set is then used to build the DPNst model: a long short-term memory network learns the user's behavior; a convolutional neural network learns the spatial relationships between the origin and the candidate destinations; and a fully connected neural network learns the external features. In the final step, DPNst dynamically aggregates the output of the three neural networks based on the given data and generates the predictions. In a series of experiments on real-world stationless bike-sharing data, DPNst returned an F1 score of 42.71% and demonstrated better performance overall than the compared baselines.

1. Introduction

Cycling is a low-carbon, environment-friendly method of transportation, and bike-sharing is the newest iteration of this popular and healthy mode of travel. Bike-sharing is based on the sharing economy, which means a community rents or shares access to good or services through online transactions. The widespread popularity of bike-sharing can be attributed to several key advantages: (1) renting and returning a bike at the roadside is convenient and affordable; (2) it can solve the last mile problem common to most mass-transit systems; and (3) it helps to alleviate traffic congestion. In fact, China's craze for bike-sharing has brought more than 2 million new bikes to its city streets [1]. In fact, Mobike, the world's largest bike operator, recently made Shanghai the world's largest bike-sharing city [2].

Unlike most other bike-sharing schemes around the world, China's shared bicycles can be picked up or dropped off anywhere; i.e., the systems are stationless. Each bike contains a GPS/3G module and an intelligent lock. Bicycles are locked by the rider after use and unlocked by the next rider by scanning a QR code on the frame using a mobile app. The app also records the user's riding history along with other data (see Figure 1).

Bike-sharing was invented in China and, while it may be convenient for users, it can be frustrating for city authorities. One of the major concerns is piled-up bikes on the sides of city streets. For example, Shanghai leads the world with 450,000 shared bicycles, nearly all of which have appeared in the past six months [1]. Beyond the traffic and pedestrian congestion problems too many bikes can cause, they are also symptomatic of an oversupply of bikes in one location, which often means a lack of bikes at another. The main cause of this problem is mobility, i.e., one-way bike use. Passengers rent a bike from one place and ride it to another place, but rarely return it to where they started. Operators can not necessarily



(a) A group of stationless shared bikes (b) Unlocking a bike with the app (c) Route history

Figure 1: Stationless bike-sharing.

(a) Excess bikes (b) Disorderly bikes (c) Bike damage

Figure 2: Problems with bike-sharing.

redistribute stocks in time to meet demand, which results in imbalances across the system. In addition, damage to bikes is fairly common, and these bikes need to be replaced to meet functional demand. Figure 2 illustrates some of these problems.

In response to the above concerns, several scholars have already explored some aspects of traffic flow and demand prediction in bike-sharing systems. For example, Bao et al. [2] used traffic trajectories to address bike lane planning problems. However, as yet, no studies have examined the back-end administrative issues associated with real-time cyclist behavior. To address this challenge, this paper presents a neural network that predicts cyclists' destinations. The ability to forecast likely destinations across a bike-sharing network would not only help companies with dispatch and reallocation but could also guide cyclists to park their bikes in the appropriate position. Further, such a system could help governments to supervise traffic, alleviate road congestion, and better plan urban construction projects.

The major contributions of this paper are summarized below:

(i) A method for generating likely candidate destinations in a stationless bike-sharing system. A set of candidate destinations is generated by mining frequent items with the FP-Growth algorithm. Historical user data is analyzed to identify the most likely destinations for each cyclist based on origin-destination itemsets, and these sets are used to train three destination prediction networks. This technique greatly simplifies the computational complexity of the model.

(ii) An innovative deep learning model to predict cyclists' destinations. The model, called DPNst, comprises three steps: (1) data preprocessing and candidate generation; (2) model construction; and (3) prediction. To build the model, user behavior is learned through a long short-term memory (LSTM) network [3], the spatial relationships between the origin and destination maps are learned through a convolutional neural network (CNN) [4], and the external features are learned through a fully connected neural network (FCNN) [5]. The final predictions are based on a dynamic aggregate of the output of these three neural networks.

(iii) A series of experiments that verify DPNst's performance on real-world data from Mobike's stationless bike-sharing system. The results show better performance.

The remainder of this paper is organized as follows. Section 2 establishes the problem definition and provides an overview of the model. The preprocessing methods are introduced in Section 3. Section 4 describes the method for generating the candidate destination set based on frequent item mining. The DPNst is presented in Section 5, followed by the experimental evaluation in Section 6. Related work is summarized in Section 7, and Section 8 concludes the paper.

2. Overview

This section begins by defining the problem of predicting destinations in a bike-sharing system, followed by an overview of the model's framework. The notations are defined in Table 1.

2.1. Problem Definition. Given a specific user u, time t, origin a, meteorological information m, and other external

Table 1: Notations in our paper.

Notation	Description
C	the candidate generation set
S	the minimum support of FP-Growth algorithm
$\mathbf{X}_u$	the input of user behavior sequence component trained by LSTMs
$\mathbf{X}_p$	the input of position map sequence component trained by CNNs
$\mathbf{X}_e$	the input of external feature sequence component trained by FCs
$\mathbf{X}_{ubs}$	the output of user behavior sequence component trained by LSTMs
$\mathbf{X}_{pm}$	the output of position map component trained by CNNs
$\mathbf{X}_{ef}$	the output of external feature component trained by FCs
$\widehat{\mathbf{X}}$	the output of the whole networks

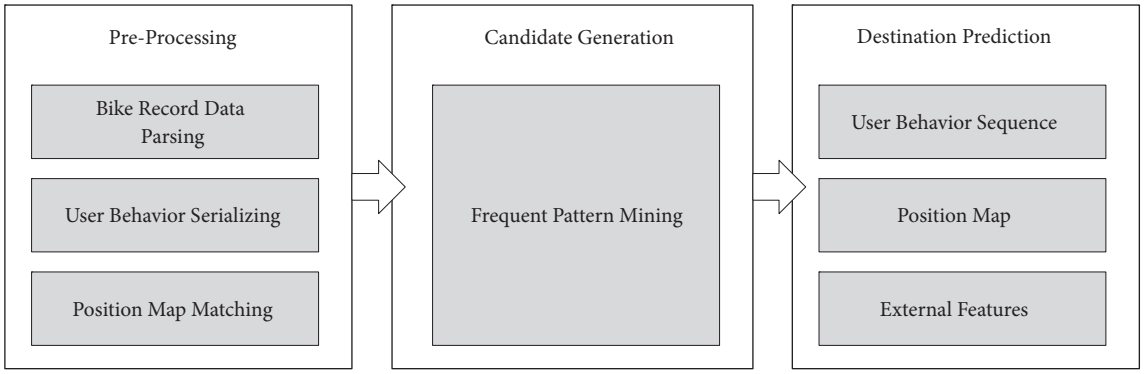


Figure 3: An overview of the framework.

information e, the probability that a bike will end its journey at destination b can be as follows.

$$P_b = f(u, t, a, m, e) \tag{1}$$

Given a set of destinations D, the predicted destination of a bicycle is the destination with the maximum probability in the following set.

$$\max_{i \in D} P_i \tag{2}$$

However, if all possible destinations were included in D, the computational complexity could be extremely high, yet ensuring that the set contains an appropriate selection of potential destinations is of great importance to the problem. This procedure is called candidate generation. Given a specific user u, time t, origin a, and a full set of positions H that includes both origins and destinations, we need to generate a manageable set of candidate likely destinations C_a^u as follows.

$$C_a^{u,t} = \{b \mid \exists a \longrightarrow b,\ User = u,\ Time = t\} \quad a, b \in H \tag{3}$$

Hence, this destination prediction problem has been converted into a recommendation problem, and finding a solution becomes a binary classification problem. If the user is likely to ride to a destination, the position is labeled with a 1, and 0 otherwise. If the user has never been to a place, the destination prediction model refers to other nearby places. Thus, generating candidates can be seen as a frequent item mining problem with the goal of identifying the most likely origin-destination itemsets for a given user.

2.2. Model Framework. The model's framework is presented in Figure 3. It consists of three main components: preprocessing, candidate generation, and destination prediction.

Preprocessing. This component is designed to process the input information, which includes bike records, map information, and meteorological information. This component has several functions: (1) to parse the bike record data and remove any outliers; (2) to match the map positions, identify the origins and destinations on the map, and extract the spatial information; and (3) to serialize the user's behavior, which converts the user's riding history into a serial format.

Candidate Generation. This component identifies the set of most likely destination candidates using the frequent pattern mining methods outlined in Section 3.

Destination Prediction. In this component, the spatial information, the user's behavior series, and external features are used to predict the user's destination from the candidates in the set. More details are provided in Section 3.

3. Preprocessing

Before constructing the model, the data needs to be preprocessed to remove as much erroneous, abnormal, and redundant data as possible to make it easier to construct a robust prediction model. However, this process must preserve the reliability and quality of the data without changing the

data distribution. Hence, the procedure involves three tasks to prepare the data for further processing.

Bike Record Data Parsing. This step filters noisy data out of the dataset. Redundant records are removed. For example, if one user has multiple records covering the same period of time, the most likely route taken is retained and the others are discarded. Incomplete records are removed, such as those missing a user ID, time, origin, destination, and so on. Records with a short distance between the origin and the destination but with a long duration are regarded as invalid and are also removed. In addition, we also found some records with latitudes and longitude beyond the range of Beijing, which were removed, along with some other outlier data found using the detection methods in [6].

Once the data is cleaned, the dataset is converted from its original field format to the input format required by the training model. For the candidate generation model, we simply extract the order ID, user ID, origin, and destination information. Then, we can get the frequent items from these datasets. However, the destination prediction model requires features such as the user behavior sequence and the position map, which demands a more complex extraction process. This is described in Section 5.

Position Mapping. In this step, the latitudes and longitudes of each position are plotted onto a corresponding map. In the Mobike dataset, each location is geohashed; therefore the hashed positions needed to be decoded into latitudes and longitudes. Once we finish mapping the positions to a matrix and map, relevant features for these positions can be extracted, such as the type of location or the local weather conditions. These features are important for constructing the spatial and external feature vectors in the following models.

User Behavior Serializing. In this step, user behaviors are sorted into a series according to time. Specific users at specific times with specific origins are converted from per-line data into a corresponding behavior sequence vector as a convenient input for the subsequent prediction models. This process is applied to every user and record. More details are provided in Section 5.1.

4. Candidate Generation

The next step is to generate a pool of candidate destinations. As previously mentioned, we have framed this destination prediction problem as a recommendation problem and the solution as a binary classification problem, where the positive samples in the training set are the ground truth destinations. However, an appropriate balance between positive and negative samples is crucial. Too many negative samples would lead to a massive computational overhead. And too many positive samples will cause the imbalance of positive and negative samples and may result in prediction failure. For example, consider a city with 10,000 possible locations but only one likely destination for a specific user at a specific time given their starting point. This would result in a 1:10,000 ratio of positive to negative samples. Multiply that by a thousand orders and the number of samples becomes $1000*10{,}000 = 10{,}000{,}000$. Thus, every additional order would increase the amount of data exponentially. However, if only the 10 most likely destinations for a thousand orders were included in the candidate pool, the number of input datasets would be just $1000*10 = 10{,}000$, which vastly reduces the computational complexity.

Hence, generating a manageable candidate pool can be seen as a frequent itemset mining problem, where the goal is to identify the most common origin-destination itemsets from a user's historical data. This is discussed in more detail in the next section.

4.1. FP-Growth. To identify the most likely user destinations, we use an approach based on frequent-pattern-trees (FP-trees), i.e., the FP-Growth algorithm [7]. FP-trees are an extended prefix tree structure for storing crucial information about frequent patterns in a compact way, and FP-Growth is an efficient FP-tree based mining algorithm for mining complete sets of frequent patterns according to pattern fragment growth.

FP-Growth first compresses the input datasets, creating an FP-tree instance to represent frequent items. Then, the compressed datasets are divided into subsets of conditional datasets, each one associated with a unique frequent pattern. Each conditional dataset is then mined separately. Using this strategy, FP-Growth not only reduces the search costs, by recursively looking for shorter patterns and concatenating them into longer frequent patterns once found, but also offers good selectivity. In this problem setting, mining frequent items from historical user data with a traditional statistical method, such as the Apriori algorithm, would be both computationally intensive and, likely, less accurate. The FP-Growth algorithm, however, can extract frequent items quickly with less overhead, making it a suitable choice for identifying the most likely user destinations for the candidate set. The next section explains the candidate generation model in more detail.

4.2. Candidate Generation Model. As Algorithm 1 shows, four different itemsets are mined from the user's historical data to construct the pool of candidate destinations. Each is explained below.

User-Origin-Destination, denoted as C_{UOD}, reflects the destinations a user has most commonly traveled to with considering the origin.

User-Origin, denoted as C_{UO}, represents all the locations where a user most frequently begins their journey without considering their destination. This itemset has been included because cyclists sometimes travel a route in reverse and the origin becomes the destination.

User-Destination, denoted as C_{UD}, reflects all the locations where a user has most often returned a bike, i.e., past destinations, because users often return to the same destinations.

Origin-Destination, denoted as C_{OD}, considers all users, not just a specific user, and reflects the most common destinations for a given starting point.

Input: Training set T
Output: Candidate Set C_{FP}
1: Initialize the time ranges t in training set T
2: Select the data items from training set T at time t
3: Get C_{UOD} according to users, origins and destinations with FPG at minimum support S_{UOD}
4: Get C_{UO} according to users and origins with FPG at minimum support S_{UO}
5: Get C_{UD} according to users and destination positions with FPG at minimum support S_{UD}
6: Get C_{OD} according to origins and destination positions with FPG at minimum support S_{OD}
7: Get the final candidates $C_{FP} = C_{UOD} \cup C_{UO} \cup C_{UD} \cup C_{OD}$
8: **Return** the candidate set C_{FP}

Algorithm 1: Candidate generation algorithm.

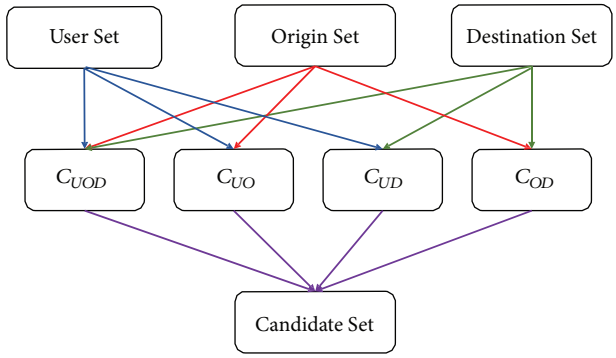


Figure 4: The candidate generation process.

With these four frequent itemsets extracted, the set of candidate destinations is constructed as follows.

$$C_{FP} = C_{UOD} \cup C_{UO} \cup C_{UD} \cup C_{OD} \tag{4}$$

Figure 4 provides a schematic overview of the process, and the algorithm is presented as Algorithm 1.

5. Destination Prediction

With the candidate destination set in hand, the next goal is to classify the likelihood that a user intends to travel to each location. Given that this is a binary classification problem, a candidate destination is labeled 1 if the destination is likely, and 0 otherwise.

5.1. Influence Factors. First of all, we need to analyze the factors that influence the users' cycling and the pattern of origins and destinations.

Users' Behavior Analysis. We use user id 2730 as an example to analyze the influence of user's behavior to the destination. As is shown in Tables 2 and 3, it can be seen from the data in May 14th that places "wx4gn0q" and "wx4gn2" appeared two times from May 11th to 13th. So, the high frequency location in historical data may be one of the destinations.

Spatial Relationship. As is shown in the Table 4, we have counted the high frequency top 10 origin-destination points. It can be found that there are strong correlations between these points and users often cycle between them. Therefore, it is necessary to learn these rules from a model.

External Factors. There are many external factors affecting traffic flow, such as weather, temperatures, and user's features. These factors have been described in [8]. Here, we use the conclusion to build models directly to learn these features.

5.2. Destination Prediction Network. Above all, the destinations are all affected by origin, the historical behavior of users, and the external features. Inspired by these factors, the model provides a detailed description of the classification tasks and the different factors considered by the three separate neural networks.

DPNst consists of three major components, as illustrated in Figure 5: a user behavior sequence model, a position map, and external features. The external features include meteorological information, riding time, and geographical features.

A user's historical behavior is first sorted into a series according to time and then is input into an LSTM network to learn the temporal rule of the origin and destination. Next, the spatial relationships between the origins and destinations are extracted and placed on a position map. This map is converted into a 2-channel image-like matrix to train the CNN and learn the spatial relationships. Lastly, the external features are input into the FCNN. The outputs of the three components are combined to produce the final results.

We adopt the parametric-matrix-based fusion method proposed in the ST-ResNet [8]. DPNst is to fuse the output from each of the component neural networks in a parametric matrix, as shown below:

$$\widehat{\mathbf{X}} = \mathbf{W}_{ubs} \circ \mathbf{X}_{ubs} + \mathbf{W}_{pm} \circ \mathbf{X}_{pm} + \mathbf{W}_{ef} \circ \mathbf{X}_{ef} \tag{5}$$

where $\circ$ is Hadamard product (i.e., element-wise multiplication) and $\mathbf{W}_{ubs}$, $\mathbf{W}_{pm}$, and $\mathbf{W}_{ef}$ are learnable parameters that adjust the degree of influence of each of the neural networks, the LSTM, the CNN, and the FCNN, respectively.

Table 2: Behaviors of user 2730 in 14th May.

user ID	time	origin	destination
3093685	2017-05-14 15:23:01	wx4gn29	wx4gn0k
2178747	2017-05-14 15:37:23	wx4gn0m	wx4gn0h
3409017	2017-05-14 17:08:20	wx4gn2h	wx4gn0r
3192545	2017-05-14 10:29:06	wx4gn21	wx4gn22
366384	2017-05-14 10:35:58	wx4gn21	wx4gn0e
164139	2017-05-14 14:40:00	wx4gn29	wx4gn2h
1682231	2017-05-14 17:40:01	wx4gn0q	wx4gn2h
3076183	2017-05-14 16:00:50	wx4gn0q	wx4gn0j
1682232	2017-05-14 21:26:15	wx4gn2h	wx4fypy
3850094	2017-05-14 16:26:45	wx4gn0q	wx4gn2h
3900595	2017-05-14 17:19:20	wx4gn0r	wx4gn0y
3093686	2017-05-14 22:00:48	wx4gn25	wx4gn29

Table 3: Behaviors of user 2730 in 11th to 13th May.

user ID	time	origin	destination
3218948	2017-05-12 21:48:31	wx4gn2m	wx4fyrf
1161301	2017-05-12 22:32:51	wx4gn2g	wx4gn2h
3530242	2017-05-12 15:18:11	wx4dzyz	wx4dzzj
2075155	2017-05-12 15:26:31	wx4dzzm	wx4epb8
94241	2017-05-11 18:57:00	wx4gn0q	wx4gn2h
759273	2017-05-12 18:18:42	wx4gn0m	wx4fyru
685779	2017-05-12 21:01:32	wx4gn0q	wx4gn2h
3622192	2017-05-13 19:54:23	wx4gn25	wx4gn0q
1229376	2017-05-13 20:16:54	wx4gn0r	wx4gn0w

Table 4: The top 10 origin and destination patterns.

origin	destination	count
wx4f9ky	wx4f9mk	681
wx4f9mk	wx4f9ky	497
wx4f9kn	wx4f9mk	437
wx4f9kn	wx4f9ms	372
wx4fg87	wx4ferq	356
wx4f9ky	wx4f9ms	356
wx4f9wb	wx4f9mu	355
wx4f9ms	wx4f9kn	345
wx4eq0c	wx4eq23	323
wx4f9mk	wx4f9kn	319

A cross combination softmax function generates the probability value of the classification prediction. The cross-entropy loss function is as follows.

$$L_\theta = -\frac{1}{m}\sum_{i}^{m} y^{(i)} \log\left(h_\theta\left(x^{(i)}\right)\right) + \left(1 - y^{(i)}\right)\log\left(1 - h_\theta\left(x^{(i)}\right)\right) \tag{6}$$

The learning process of the DPNst is shown in Algorithm 2. We first construct the training instances from the datasets. Then, DPNst is trained via backpropagation [9] and Adam [10].

5.3. The Structure of User Behavior Sequence Component. Given that bike-sharing users repeatedly rent bikes over a period of time, their historical data can be formulated as a time series, i.e., a behavior sequence with a time attribute. Typically, time series data are trained with a recurrent neural network (RNN) [11]. However, in recent years, LSTMs [3] have been successfully used to train complex time series data in a variety of applications, such as highway traffic prediction [12], traffic speed prediction [13], and tourism prediction [14]. Unlike the simple neurons in an RNN, LSTM neurons contain an input gate, an output gate, a cell, and a forget gate that determines how the information flows into and out of the neuron. Moreover, because LSTMs were specifically developed to overcome the exploding and vanishing gradient problems associated with training traditional RNNs in some scenarios, LSTMs are particularly well-suited to classification, processing, and prediction tasks with time series data that contain a time lag between important events of an unknown size or duration. Hence, the user behavior sequence component in DPNst is based on an LSTM network.

First, the data is converted into a sequence of user behaviors according to time, and the number of user-destination itemsets is counted to generate a sequence of behavior according to time. Assuming the current moment

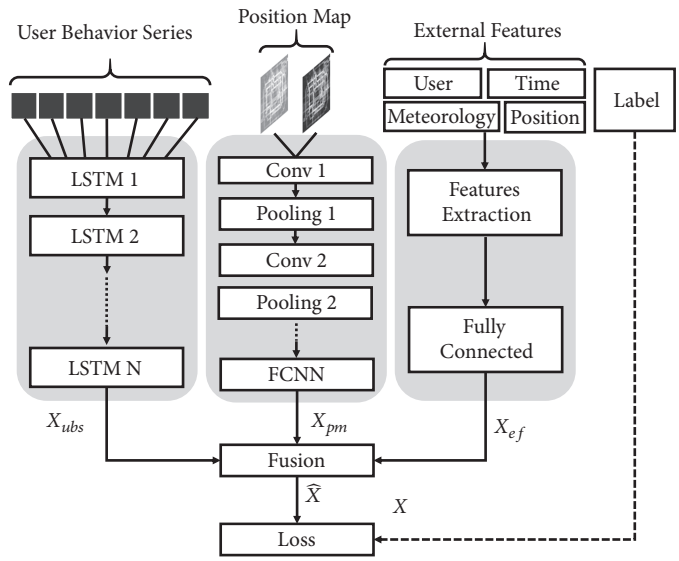


Figure 5: The network architecture.

Input: construct $\mathbf{X}_u = \{x_1, x_2, ..., x_{n-1}\}$, $\mathbf{X}_p = \{x_{ori}, x_{dest}\}$, $\mathbf{X}_e = \{x_1, x_2, ..., x_{14}\}$ from candidate set C_{FP}
Output: Learned Destination Prediction Model f
1: Initial the parameters θ in the networks
2: **Repeat**
3: input $\mathbf{X}_u$ into the LSTM and get $\mathbf{X}_{ubs} = f_u(\mathbf{X}_u)$
4: input $\mathbf{X}_p$ into the CNN and get $\mathbf{X}_{pm} = f_p(\mathbf{X}_p)$
5: input $\mathbf{X}_e$ into the FCNN and get $\mathbf{X}_{ef} = f_e(\mathbf{X}_e)$
6: find the best θ with a cross-entropy loss function
7: **Until** get the best $\widehat{X} = f(\mathbf{X}_{ubs}, \mathbf{X}_{pm}, \mathbf{X}_{ef})$

Algorithm 2: Destination prediction training algorithm.

in time is t and the number of time windows is n, these sequences are constructed as $[x_{t-n}, x_{t-(n-1)}, \ldots, x_{t-1}]$, which represents the total number of cycling trips from the origin to the destination in each time window. If there are no recorded trips in a window, the value is 0.

To identify and extract these patterns from historical behavior, a LSTM with many layers and hidden units is needed, as shown in Figure 6. A sequence is input into the first layer of the LSTM and output through a series of hidden units to the next layer. The final output is the output of the last hidden unit in the last LSTM layer. That output is then fed into a softmax activation function to generate the final prediction result.

For hidden unit h at time t, the output of h is presented as

$$
\begin{aligned}
f_t &= \sigma\left(W_f\left[h_{t-1}, x_t\right] + b_f\right) \\
i_t &= \sigma\left(W_i\left[h_{t-1}, x_t\right] + b_i\right) \\
\widetilde{C}_t &= \tanh\left(W_c\left[h_{t-1}, x_t\right] + b_C\right) \\
C_t &= f_t * C_{t-1} + i_t * \widetilde{C}_t \\
o_t &= \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right) \\
h_t &= o_t * \tanh\left(C_t\right)
\end{aligned}
\tag{7}
$$

where W denotes the weight matrixes, b denotes the bias vectors, i represents the input gate, f represents the forget gate, and o represents the output gate; σ is a sigmoid function.

The final prediction result is

$$
\mathbf{X}_{ubs} = W * h_t + b \tag{8}
$$

where h is the last hidden unit.

5.4. The Structure of Position Map Component. The biggest difference between predicting user behavior in a bike-sharing scenario and traditional time series problems is that bike-sharing data has spatiotemporal qualities. Therefore, capturing the relationships between spatial positions is very important. The relationship between spatial positions can be mapped as a two-dimensional matrix, which can, in turn, be regarded as a graph. Hence, the relationship between one point and another can be seen as the relationship between different geographical locations.

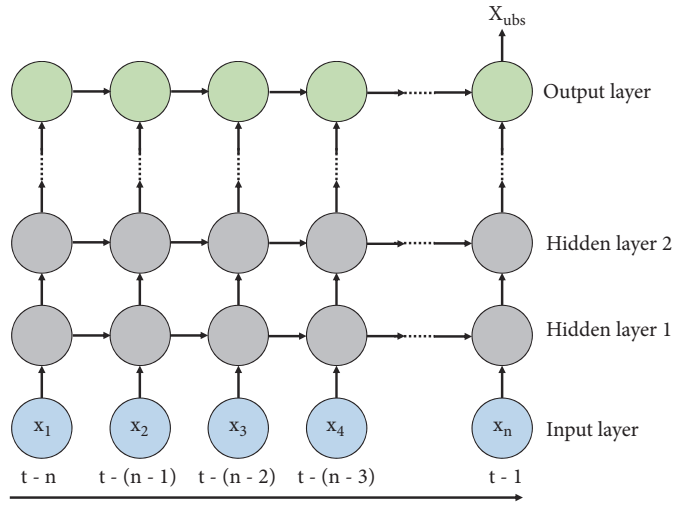


Figure 6: The architecture of the LSTM.

CNNs [4, 15] are well-suited to dealing with image information and can flexibly capture local relationships within and between images. Further, CNNs have a proven and powerful ability to hierarchically capture structural spatial information while extracting key features, and convolutional operations can be pooled to reduce complexity. Hence, CNNs constitute a highly appropriate way to extract information about the relationships between spatial locations on a map.

The first step in this component is to accurately place all the candidate destinations as positions on a map, so the CNN can extract the relationships between each position. However, given that one convolutional layer can only consider near spatial dependencies, as limited by the size of the kernels [8], the CNN in DPNst needs to contain several convolutional and pooling layers, as shown in Figure 7.

The origin and candidate destinations are parsed as a 2-channel map. Each position is marked on a 2D image; the origin is labeled as 1; the destinations are labeled 0. We handle all the datasets into maps and convert them into a tensor $\mathbf{X}^{(0)} \in \mathbb{R}^{r*I*J*2}$ where r is the numbers of maps, I is the height of the maps, and J is the width of the maps. A convolution layer follows, expressed as

$$\mathbf{X}_c^1 = f\left(W_c^1 * X_c^0 + b_c^1\right) \tag{9}$$

where $*$ denotes the first layer of the CNN, f is an activation function, e.g., ReLU, and W_c^1, b_c^1 are the learnable parameters in the first layer. Then, $\mathbf{X}_c^1$ is input into the pooling function F as follows.

$$\mathbf{X}_p^1 = F\left(X_c^1\right) \tag{10}$$

In our DPNst, we stack l convolution layers and pooling layers. Through multilayer convolution, the network could find the corresponding relationship between the origin and destination of different users. Finally, we add 2-layer FCNN to get the final result $\mathbf{X}_{pm}$.

5.5. The Structure of External Component. Beyond the relationships inherent in historical user behavior and locations, other factors, such as weather and the time of day, are also important. Even a user's personal information may affect their mode of travel and their destination. Therefore, DPNst contains an FCNN to parse these external features. The features considered follow.

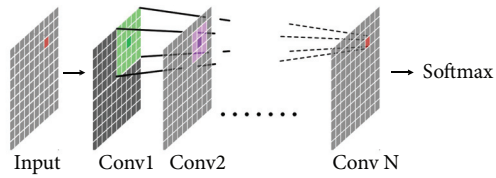


Figure 7: The architecture of the CNN (1 channel).

User Features. Each user has a unique identity in the dataset, which can be used as a basis for distinguishing specific personality traits reflected in the user's riding history. Therefore, through one-hot encoding, x_1 denotes the user ID, and the dimension represents the number of users.

Time Features. Time plays a direct role in the relationship between an origin and a destination because, naturally, users often travel to the same destination at a specific time of the day or week. Hence, the current time is constructed as a feature. We define the x_2 as month, x_3 as day, x_4 as hour, x_5 as minute, x_6 as weekday, and x_7 as weekend or public holiday (if today is not a weekday, the value is 1, else 0).

Meteorology Features. Weather affects many things, including traffic [16], a user's preferred mode of transport, and how far they are likely to ride. x_8 denotes the temperature at daytime and x_9 at night (in °C), and x_{10} denotes the Beaufort wind force scale. These features are all continuous values. x_{11} denotes the weather conditions. These values are discrete variables, such as sunny and raining. These

categorical variables are encoded into a numerical vector using one-hot encoding and allocated to a categorical length vector, as shown in (11). This method can improve a model's training performance. x_{12} represents the air quality index (AQI).

$$x_{ij} = \begin{cases} 1, & \text{if } x_i \text{ is in category } j \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

Position Features. These features represent the geographical characteristics of a location. x_{13} represents the distance between the origin and the candidate destination in terms of the Haversine equation, shown in (12). x_{14} is the location category, such as office, school, residence, and community service. Again, these features are defined through one-hot encoding:

$$\begin{aligned} h_{i,j} &= \sin^2(\Delta lat) + \cos(lat_i)\cos(lat_j) + \sin^2(\Delta lng) \\ d_{i,j} &= 2 \cdot R \cdot \arcsin\left(\sqrt{h_{i,j}}\right) \end{aligned} \tag{12}$$

where (lng_i, lat_i) and (lng_j, lat_j) are the latitude and longitude in radians of the two locations. Degrees are converted into radians by multiplying by $\pi/180$ as usual. $\Delta lat = (lat_i - lat_j)/2$, $\Delta lng = (lng_i - lng_j)/2$, and R is the radius of the Earth, which is about 6371 km.

The external features are constructed, then normalized, and input into a multilayer FCNN to learn their regularities. The final output is denoted as $\mathbf{X}_{ef}$.

6. Experiments

6.1. Datasets. To evaluate DPNst's performance, we conducted a series of experiments using datasets from Mobike's stationless bike-sharing scheme in Beijing combined with meteorological data from the Biendata Platform [17]. The Mobike dataset spans the period 10-24 May 2017 and contains over 3 million historical records for around 349,000 users and 485,000 bikes. The information includes order ID, user ID, bike ID, bike type, start time, and geohashed origins and destinations. The meteorological information spans the same time period and was sourced from the China Meteorological Administration website [18]. It includes weather conditions, temperatures, wind directions, Beaufort wind force scales, and other information. The statistics for each dataset are provided in Table 5.

6.2. Preprocessing. The data was preprocessed following the procedure outlined in Section 3. Then, we sampled the data, at different rates for each of the three neural networks, to reduce the computing complexity, ensuring that an appropriate balance between positive and negative records was maintained. The min-max normalization method was used to scale the data to the correct range [-1;1].

6.3. Baseline

6.3.1. Baseline of Candidate Generation. To evaluate each aspect of the candidate generation method, we constructed four baselines as follows.

Table 5: The details of Mobike datasets.

Datasets	Mobike Beijing
Time Period	10 May 2017 to 24 May 2017
Number of Users	349,693
Number of Bikes	485,465
Number of Records	3,214,096
Gird map size	(1452, 1716)
Range of Latitude	(20.01N, 40.66N)
Range of Longitude	(102.65E, 122.13E)
Datasets	**Meteorology**
Weather conditions	6 types (e.g., Sunny, Rainy)
Temperature / (°C)	[11, 34]
Beaufort Wind Force Scale	[2, 5]
Air Quality Index (AQI)	[40, 396]

User-Destination Count (UD). We identified the user-destination itemsets with the highest counts as candidate destinations.

User-Origin Count (UO). We scan specific users and destinations to identify the highest counting items as candidate destinations and add to UD.

Origin-Destination Count (OD). We used statistical methods to scan the origins and destinations of all users to find out the highest counting items as the candidates and add to UD and UO.

Candidate Generation Model (CGM). The most frequent itemsets for user-origin-destination were determined with the FP-Growth algorithm, using different minimum support parameters for each itemset. We will set four sets of minimum support to verify the effect of the model.

6.3.2. Baseline of Destination Prediction. Similarly, to evaluate various aspects of the destination prediction model, we constructed four further baselines as follows.

Historical Count (HC). The training set included the destinations a specific user went to the most times; the testing set included the latest data.

Naive Bayesian (NB). We use a simple naive Bayesian model to predict the destination by conditional probability by using the latest data in the training set.

To assess each of DPNst's three components, we constructed three further baselines as follows:

DPNst1: UBS. Only the LSTM was used to train the user behavior sequences.

DPNst2: UBS + PM. The LSTM was used to train the user behavior sequences and the CNN was used to train the position maps.

DPNst3-5: UBS + PM + EF. The LSTM was used to train the user behavior sequences, the CNN was used to train the position maps, and a multilayered FCNN was used to train the external features.

6.4. Hyperparameters. All models were built using Python libraries, including Numpy, Pandas, scikit-learn, and Tensorflow [19] (GPU version 1.2.1). Descriptions of the hyperparameters for both the CGM and DPNst models follow.

Hyperparameters of CGM. The FP-Growth algorithm within the CGM includes three defined hyperparameters. Minimum support S_{UOD} gauges the correlations between frequent user, origin, and destination items. Minimum support S_{UO} gauges the confidence in the correlation between frequent user and origin items. Minimum support S_{UD} gauges the correlations between frequent user and destination items. And minimum support S_{OD} gauges the correlations between frequent origin and destination items. The appropriate levels of minimum support are tuned through experimentation. The smaller the support, the higher the recall and the higher the mean numbers of candidate. So, we need to find the values that could keep balance.

Hyperparameters of DPNst. The LSTM contains 10 hidden units, with a variable number of layers. In the CNN, Conv1 contains two $5 * 5$ filters, and Conv2 contains four $10 * 10$ filters, each with a batch size of 1000. The drop-out rate was set to 0.8. The model was subsequently trained on the full set of training data for a fixed number of epochs. However, it is worth noting that hardware configurations significantly affect the optimal parameter settings. Therefore, these parameters need to be tuned to suit the specific platform configuration. The increase of LSTM layers number could learn more users behavior from the data and the increase of FCNN layers could also learn more from the external features. But the layers of CNN may not need to be higher than 3; it would be more computing cost.

6.5. Evaluation Metrics

6.5.1. Baseline of Candidate Generation. We used recall to evaluate the performance of both the candidate generation and destination prediction model, and the mean number of candidate destinations to evaluate the candidate generation model. The formulas for each metric follow:

$$Recall = \frac{TP}{TP + FN}$$
$$Mean = \frac{1}{N}\sum N_i \quad (13)$$

where TP are the true positive samples, FN are the false negative samples, and N_i is the number of candidate destinations in the ith sample.

6.5.2. Baseline of Destination Prediction. Additionally, we used F1-scores and accuracy to evaluate the performance of the destination prediction model. The formulas follow:

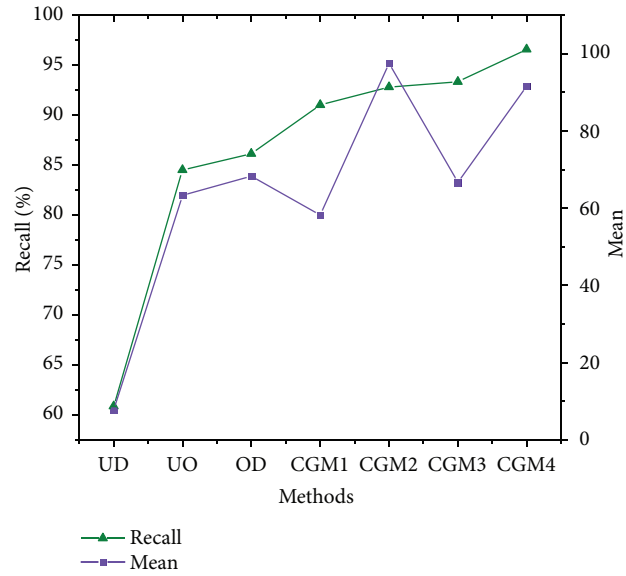


Figure 8: The candidate generation results.

$$Precision = \frac{TP}{TP + FP}$$
$$F1 = 2\frac{Precision \cdot Recall}{Precision + Recall} \quad (14)$$

where FP are the false positive samples.

6.6. Results

6.6.1. Results of Candidate Generation. The experimental results are provided in Table 6 and Figure 8, where the results generated by each of different baselines are clear. The baseline relying solely on the frequent user-destination itemset had the lowest minimum recall at 60.89% with a mean of 7.76. Although the mean is small, the cover rate is quite low. However, after including the frequent users-origin itemsets, recall increased to 84.51% with a mean of 63.38. Moreover, after including the frequent origin-destination itemsets, recall increased even further to 86.13% with a sharp increase in the mean to 68.31.

The results for the full candidate generation model show a still more significant improvement. Recall increased to 91.02%, and the mean decreased to 58.29 with a minimum support of $S_{USD} = 1$, $S_{US} = 2$, $S_{UD} = 2$, $S_{SD} = 3$. And, as the minimum support decreased, both the recall and the mean increased. This is because decreasing the support relaxes the constraints on the frequent itemsets and more itemsets become available to satisfy the condition. This result also demonstrates the importance of choosing the appropriate parameters to balance the demands of computational complexity with the desired accuracy of the prediction model. Unfortunately, there is no standard choice and the parameters need to be tuned for each specific hardware configuration. In this paper, we chose $S_{USD} = 1$, $S_{US} = 2$, $S_{UD} = 2$, and $S_{SD} = 2$ as the best solution because these settings produced very high recall with an acceptable mean.

Table 6: Results of candidate generation.

Methods	Recall	Mean
User-Destination Count (UD)	60.89%	7.76
User-Origin Count (UO)	84.51%	63.38
Origin-Destination Count (OD)	86.13%	68.31
Our Methods		
Candidate Generation Model1 (CGM1)		
$S_{USD} = 1, S_{US} = 2, S_{UD} = 2, S_{SD} = 3$	91.02%	58.29
Candidate Generation Model2 (CGM2)		
$S_{USD} = 1, S_{US} = 1, S_{UD} = 1, S_{SD} = 3$	92.81%	97.57
Candidate Generation Model3 (CGM3)		
$S_{USD} = 1, S_{US} = 2, S_{UD} = 2, S_{SD} = 2$	**93.34%**	**66.70**
Candidate Generation Model4 (CGM4)		
$S_{USD} = 1, S_{US} = 2, S_{UD} = 2, S_{SD} = 1$	96.57%	91.58

Table 7: Results for different methods of destination prediction.

Methods	Recall	Precision	F1
Historical Count (HC)	28.96%	24.95%	26.81%
Naive Bayesian (NB)	32.14%	27.69%	29.75%
Our Methods (l means layers)			
DPNst1: UBS			
(2-l LSTM)	33.12%	27.58%	30.10%
DPNst2: UBS + PM			
(2-l LSTM + 2-l CNN)	35.46%	30.55%	32.82%
DPNst3: UBS + PM + EF			
(2-l LSTM + 2-l CNN + 2-l FCNN)	37.54%	32.34%	34.75%
DPNst4: UBS + PM + EF			
(2-l LSTM + 2-l CNN + 5-l FCNN)	31.98%	59.56%	41.62%
DPNst5: UBS + PM + EF			
(5-l LSTM + 2-l CNN + 5-l FCNN)	**35.27%**	**54.12%**	**42.71%**

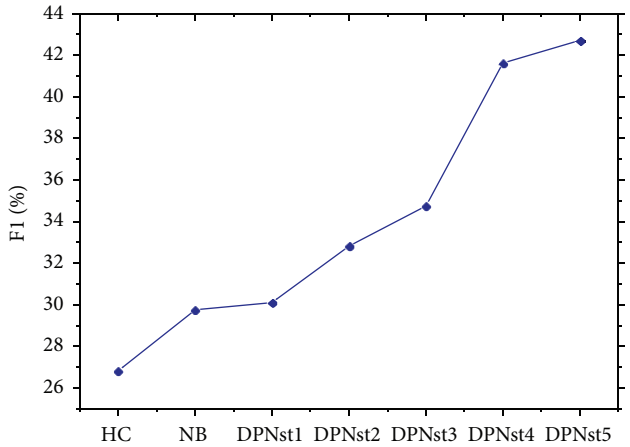


Figure 9: The destination prediction results.

6.6.2. Results of Destination Prediction. The results from the destination prediction evaluation are presented in Table 7 and Figure 9 which, again, clearly show the effect of each baseline. The baseline relying only on the highest position count resulted in the least accuracy followed by the naive Bayesian model. DPNst was the most accurate, demonstrating a 15.9% increase over the next best approach in terms of F1. DPNst was the most accurate, with a recall of 35.27%, a precision of 54.12%, and an F1 score of 42.71%, representing an increase of 15.9% in terms of F1. These results provide support for DPNst as a well-performing model in bike-sharing systems.

To further assess the model, we analyzed its performance at the component and layer level. As each 2-layer network was added, the recall, precision, and F1 scores increased. However, with a 5-layer FCNN, recall decreased to 31.98%, yet precision increased to 59.56%, and the F1 score increased to 41.62%. The best performance resulted from adding a 5-layer LSTM where recall increased to 35.27%, precision decreased to 54.12%, and the F1 score increased to 42.71%. This confirms that each component of DPNst helps to improve the model's performance.

7. Related Work

In recent years, bike-sharing has received increasing attention due to its significance as an environmentally friendly form

of travel and its ability to overcome the "last mile" problems associated with other forms of mass transit. Studies on bike-sharing span both station-based and stationless systems, with many focusing on public schemes. DeMaio [20] provided an introduction to bicycle-sharing systems, including their history, impacts, models, and what the future for research in this field may hold. Etienne et al. [21] studied a statistical model of public bicycle travel based on Velib's system in Paris, France. Midgley [22] analyzed the state-of-the-art and users' experiences with several station-based public bicycle systems across Europe. These early studies laid the foundational concepts and working mechanisms of public bike-sharing systems.

In time, scholars began to examine some of the problems associated with bike-sharing schemes. Given that people's bike use tends to be quite skewed and imbalanced, Pavone et al. [23] developed methods for maximizing the throughput of a mobility-on-demand urban transportation system and introduced a rebalancing policy to minimize the number of vehicles needed for rebalancing trips. This advancement provided vital inspiration for solving balance and load problems in public bike-sharing systems.

Studying user behavior patterns in bike-sharing systems helped us to understand the flow and mobility patterns in public bicycle traffic. For example, Jon Froehlich et al. [24] presented a 13-week spatiotemporal analysis of bicycle station usage in Barcelona's bike-sharing system. Kaltenbrunner et al. [25] provided an analysis of human mobility data in urban areas based on the number of available bikes at Bicing stations C, a community bicycle program in Barcelona. Vogel et al. [26] adopted clustering and validation to analyze bike usage patterns in Vienna. Beyond insights into mobility and public bicycle flow, these studies also contributed the notion of leveraging station clustering, based on geographical position and transition patterns, as a way to reallocate bicycles to compensate for imbalanced use.

Contardo et al. [27] and Benchimol et al. [28] each presented mathematical formulations to reroute vehicles and transfer bikes. These formulations consider external features, such as vehicle capacity and the extent of the imbalance. However, simply monitoring the current number of bikes at each station and reallocating bikes after an imbalance occurs constitute a remedy for the problem, not a cure. Hence, a new set of studies that explored ways to predict potential imbalances in advance emerged.

Borgnat et al. [29] used a combination model and Velov's dataset to predict traffic across the entire bike-sharing system at each hour of the day. Vogel et al. [26, 30] used time series analysis to forecast bike demand in Vienna, while Yoon et al. [31] used a modified ARIMA model to predict the available bikes and docks at each station by considering temporal and spatial factors. These studies provided insights into the influence of traditional market impacts on bike-sharing systems. Zheng et al. [32] predicted traffic flows at the check-in and check-out areas of New York and Washington's public bicycle systems from a macro perspective, contributing a clustering algorithm based on k-means and a transition matrix. Conversely, Zhang et al. [33] adopted a micro perspective, using GBRT and Lasso regression to predict user behavior and travel times in Chicago's public bicycle-sharing system. Each of these studies focussed on prediction: available bikes and docks, passenger numbers and flow at check-ins and check-outs, and so on. Their studies are the most closely related to the destination prediction problem explored in this paper.

Because stationless bike-sharing is a relatively newer business model, less research has been undertaken in this area. Jie [2] presented a data-driven approach for developing bike lane construction plans in Shanghai based on Mobike's stationless trajectory data. This study examined mobility statistics in Mobike's data, providing much of the inspiration for applying data mining techniques.

Further, some existing research has already been conducted on destination prediction. Natalia and Chris [34] and Patterson et al. [35] both used a Bayesian method to predict destinations for specific individuals based on their historical modes of transport. Tiesyte and Jensen [36] proposed a nearest-neighbor trajectory method that uses distance measures to identify the historical trajectory most similar to the current partial trajectory. Chen et al. [37] used a tree structure to represent historical movement patterns, which can be matched to the current partial trajectory by stepping down the tree. Zhang et al. [38] employed a Bayesian framework to model the distribution of a user's destination based on their travel history using the DiDi Taxi dataset. Whereas each of these studies focuses on a traditional method, such as Bayesian frameworks or a nearest-neighbor method, our work incorporates deep learning to construct more intelligent models for destination prediction. Tang et al. [39] presented a system called PARecommender, which predicts traffic conditions and provides route recommendation based on generated traffic patterns.

Deep learning is an emerging, but already widely studied, field. CNNs have been successfully applied to many different kinds of problems, especially in the field of computer vision [15]. Further, as a quasi-substitute for CNNs, the capsule network was introduced Sabour et al. [40], which is a group of neurons with an activity vector that represents the instantiation parameters of a specific type of entity, such as an object or an object part. RNNs have found success in sequencing learning tasks [11], while other types of new networks have emerged to deal with spatiotemporal data. By extending a fully connected LSTM (FC-LSTM) to incorporate convolutional structures in both the input-to-state and state-to-state transitions, Shi et al. [41] proposed the convolutional LSTM (ConvLSTM) and used it to build an end-to-end trainable model for the precipitation nowcasting problem. These networks and architectures contributed to many of our ideas for constructing a destination prediction network.

Finally, the availability of massive amounts of mobility data from users, cars, and public transport systems has given rise to many urban computing techniques that solve tasks based on real-world travel demands [42]. For example, Zheng [43] mined patterns in taxi trajectories to recommend new road construction and public transport infrastructure projects. Yuan et al. [44] used traffic patterns and POI distributions to infer the different functional zones in a city.

These studies established new research methods for dealing with transportation datasets and problems.

8. Conclusion

In this paper, we proposed an innovative deep learning model called destination prediction network based on spatiotemporal data or DPNst for short. DPNst predicts the most likely destination of a cyclist in a bike-sharing system. The first step is to preprocess the data and identify the most likely candidate destinations using frequent item pattern mining. The DPNst model is then built through a series of three neural networks using this candidate set. An LSTM network [3] learns the user behavior. A CNN [4, 15] learns the spatial relationships between the origin and the candidate destinations, and an FCNN [5] learns the external features. To the best of our knowledge, this is the first proactive method to address the administrative problems associated with bike-sharing systems by predicting the most probable user destinations. A series of experiments with real-world data from Mobike show that DPNst achieves satisfactory prediction results, with an F1 score of 42.71%, and a better performance overall than the baseline methods. In future research, we hope to improve DPNst's performance and extend the model to destination prediction for taxis, private cars, and other problems that relate to traffic destination prediction.

Data Availability

The Mobike data used to support the findings of this study have been deposited in the Biendata competition platform (https://biendata.com/competition/mobike/data/).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is partially supported by grants from the National Natural Science Foundation of China (No. 61602141) and Science and Technology Program of Zhejiang Province (No. 2018C04001).

References

[1] https://www.ft.com/content/5efe95f6-0aeb-11e7-97d1-5e720a26771b.

[2] J. Bao, T. He, S. Ruan, Y. Li, and Y. Zheng, "Planning bike lanes based on sharing-bikes' trajectories," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2017*, pp. 1377–1386, Canada, August 2017.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[4] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[5] R. J. Gibbens and F. P. Kelly, "Dynamic routing in fully connected networks," *IMA Journal of Mathematical Control and Information*, vol. 7, no. 1, pp. 77–111, 1990.

[6] Y. Zheng, "Trajectory data mining: an overview," *ACM Transactions on Intelligent Systems and Technology*, vol. 6, no. 3, article 29, 2015.

[7] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: a frequent-pattern tree approach," *Data Mining and Knowledge Discovery*, vol. 8, no. 1, pp. 53–87, 2004.

[8] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proceedings of the 31st AAAI Conference on Artificial Intelligence, AAAI 2017*, pp. 1655–1661, USA, February 2017.

[9] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," *Lecture Notes in Computer Science*, vol. 1524, pp. 9–50, 1998.

[10] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *Computer Science*, 2014.

[11] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[12] F. Altche and A. de La Fortelle, "An LSTM network for highway trajectory prediction," in *Proceedings of the 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 353–359, Yokohama, October 2017.

[13] Z. Cui, R. Ke, and Y. Wang, "Deep Bidirectional and Unidirectional LSTM Recurrent Neural Network for Network-wide Traffic Speed Prediction," 2018.

[14] Y. Li and H. Cao, "Prediction for Tourism Flow based on LSTM Neural Network," *Procedia Computer Science*, vol. 129, pp. 277–283, 2018.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[16] F. Lin, J. Jiang, J. Fan, and S. Wang, "A stacking model for variation prediction of public bicycle traffic flow," *Intelligent Data Analysis*, vol. 22, no. 4, pp. 911–933, 2018.

[17] https://biendata.com/competition/mobike/data/.

[18] http://www.cma.gov.cn/2011qxfw/2011qsjgx/.

[19] https://www.tensorflow.org/.

[20] P. Demaio, "Bike-sharing: history, impacts, models of provision, and future," *Journal of Public Transportation*, vol. 12, no. 4, pp. 41–56, 2009.

[21] E. Come and O. Latifa, "Model-Based Count Series Clustering for Bike Sharing System Usage Mining: A Case Study with the Velib' System of Paris," *Acm Transactions on Intelligent Systems and Technology*, vol. 5, no. 3, pp. 1–21, 2014.

[22] P. Midgley, "The Role of Smart Bike-sharing Systems in Urban Mobility," *Journeys*, vol. 2, no. 2, 2009.

[23] S. L. Smith, M. Pavone, M. Schwager, E. Frazzoli, and D. Rus, "Rebalancing the rebalancers: Optimally routing vehicles and drivers in mobility-on-demand systems," in *Proceedings of the 1st American Control Conference, ACC 2013*, pp. 2362–2367, IEEE, Washington, DC, USA, June 2013.

[24] J. Froehlich, J. Neumann, and N. Oliver, "Sensing and Predicting the Pulse of the City through Shared Bicycling," in *Proceedings of the 21st international jont conference on Artifical intelligence*, pp. 1420–1426, Morgan Kaufmann Publishers Inc., Pasadena, California, USA, 2009 (Bulgarian).

[25] A. Kaltenbrunner, R. Meza, J. Grivolla, J. Codina, and R. Banchs, "Urban cycles and mobility patterns: Exploring and predicting trends in a bicycle-based public transport system," *Pervasive and Mobile Computing*, vol. 6, no. 4, pp. 455–466, 2010.

[26] P. Vogel, T. Greiser, and D. C. Mattfeld, "Understanding bike-sharing systems using data mining: exploring activity patterns," *Procedia-Social and Behavioral Sciences*, vol. 20, no. 6, pp. 514–523, 2011.

[27] C. Contardo, C. Morency, and L. Rousseau, *Balancing A Dynamic Public Bike-Sharing System*, vol. 4, CIRRELT, Montreal, Canada, 2012.

[28] M. Benchimol, "Balancing the stations of a self service 'bike hire' system," *RAIRO-Operations Research*, vol. 45, no. 1, pp. 37–61, 2011.

[29] P. Borgnat, P. Abry, P. Flandrin, C. Robardet, J.-B. Rouquier, and E. Fleury, "Shared bicycles in a city: A signal processing and data analysis perspective," *Advances in Complex Systems (ACS)*, vol. 14, no. 3, pp. 415–438, 2011.

[30] P. Vogel and D. C. Mattfeld, "Strategic and Operational Planning of Bike-Sharing Systems by Data Mining – A Case Study," in *Proceedings of the International Conference on Computational Logistics*, vol. 6971, pp. 127–141, Springer Berlin Heidelberg, Hamburg, Germany.

[31] J. W. Yoon, F. Pinelli, and F. Calabrese, "Cityride: A predictive bike sharing journey advisor," in *Proceedings of the 13th International Conference on Mobile Data Management, MDM 2012*, pp. 306–311, IEEE, Karnataka, India, July 2012.

[32] Y. Li, Y. Zheng, H. Zhang, and L. Chen, "Traffic prediction in a bike-sharing system," in *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ACM, Seattle, WA, USA, 2015.

[33] J. Zhang, X. Pan, M. Li, and P. S. Yu, "Bicycle-Sharing System Analysis and Trip Prediction," in *Proceedings of the 2016 17th IEEE International Conference on Mobile Data Management (MDM)*, pp. 174–179, IEEE, Porto, Portugal, June 2016.

[34] N. Marmasse and C. Schmandt, "A user-centered location model," *Personal and Ubiquitous Computing*, vol. 6, no. 5-6, pp. 318–321, 2002.

[35] D. J. Patterson, L. Liao, D. Fox, and H. Kautz, "Inferring high-level behavior from low-level sensors," in *Proceedings of the 5th International Conference on Ubiquitous Computing*, pp. 73–89, Springer, Seattle, WA, USA, 2003.

[36] D. Tiesyte and C. S. Jensen, "Similarity-based prediction of travel times for vehicles traveling on known routes," in *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM GIS 2008*, pp. 105–114, ACM, Irvine, CA, USA, November 2008.

[37] L. Chen, M. Lv, and G. Chen, "A system for destination and future route prediction based on trajectory mining," *Pervasive and Mobile Computing*, vol. 6, no. 6, pp. 657–676, 2010.

[38] L. Zhang, T. Hu, Y. Min et al., "A taxi order dispatch model based on combinatorial optimization," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2017*, pp. 2151–2159, Canada, August 2017.

[39] F. Tang, J. Zhu, Y. Cao et al., "PARecommender: A pattern-based system for route recommendation," in *Proceedings of the 25th International Joint Conference on Artificial Intelligence, IJCAI 2016*, pp. 4272-4273, USA, July 2016.

[40] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic Routing Between Cap- sules," *NIPS Proceedings*, 2017.

[41] X. Shi, *Convolutional LSTM Network: A Machine Learning Appro- ach for Precipitation Nowcasting*, Convolutional LSTM Network, A Machine Learning Appro- ach for Precipitation Nowcasting, 2015.

[42] Y. Zheng, *Urban Computing: Concepts, Methodologies, and Appli- cations , Acm Transactions on Intelligent Systems Technology 5.3:1-55*, Acm Transactions on Intelligent Systems Technology 5.3, 1-55, 2014.

[43] Y. Zheng, Y. Liu, J. Yuan, and X. Xie, "Urban computing with taxicabs," in *Proceedings of the 13th International Conference on Ubiquitous Computing (UbiComp '11)*, pp. 89–98, ACM, September 2011.

[44] J. Yuan, Y. Zheng, and X. Xie, "Discovering regions of different functions in a city using human mobility and POIs," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '12)*, pp. 186–194, August 2012.

Hindawi
Complexity
Volume 2018, Article ID 6101409, 10 pages
https://doi.org/10.1155/2018/6101409

Research Article

Targeted Influential Nodes Selection in Location-Aware Social Networks

Susu Yang, Hui Li, and Zhongyuan Jiang

School of Cyber Engineering, Xidian University, Xi'an 710126, China

Correspondence should be addressed to Hui Li; hli@xidian.edu.cn

Received 2 September 2018; Accepted 10 October 2018; Published 1 November 2018

Guest Editor: Jianxin Li

Given a target area and a location-aware social network, the location-aware influence maximization problem aims to find a set of seed users such that the information spread from these users will reach the most users within the target area. We show that the problem is NP-hard and present an approximate algorithm framework, namely, TarIM-SF, which leverages on a popular sampling method as well as spatial filtering model working on arbitrary polygons. Besides, for the large-scale network we also present a coarsening strategy to further improve the efficiency. We theoretically show that our approximate algorithm can provide a guarantee on the seed quality. Experimental study over three real-world social networks verified the seed quality of our framework, and the coarsening-based algorithm can provide superior efficiency.

1. Introduction

In recent years, social networks have become prevalent platforms for the spread of product adoption, ideas, and news. Under this trend, influence maximization (IM) problem is becoming popular, which aims to seek k users (referred to as *seeds*) to maximize the number of influenced users (referred to as *influence spread*) in the network. Kempe et al. [1] proved that this problem is NP-hard and presented an $(1 - 1/e)$-approximate algorithm by greedily selecting k seed users which has the maximum marginal gain of influence spread. Motivated by their work, a vast amount of studies then focused on improving the effect of influence spread and the efficiency, such as the heuristic-based algorithm PMIA [2] and the sketch-based algorithm IMM [3].

However, many real-world applications such as location-based word-of-mouth marketing recently have location-aware requirements in IM. In [4], Li et al. focused on a location-aware IM (LAIM) *query*, which aims to seek k users to maximize the expected influence in a query region. They assumed that the network and the location of users are given beforehand, where an index can be constructed offline, and the target region is submitted online as a query. However, such assumption may not be always satisfied as the network and IM request are given at the same time, which is exactly the scenario discussed in traditional IM works [2, 5, 6]. Besides, existing works in LAIM can only answer the problem towards simple regions such as a rectangle or a circle. However, locations of users are always complicatedly visualized and managed via maps, the atomic regions of which are not necessarily rectangles or circles. Instead, they always show up as various and complex polygons. Therefore, it is meaningful to find an efficient method to address the LAIM problem, by targeting at an arbitrary polygon region, from scratch. Below we provide a running example to elaborate this point.

Example. A company wants to sell a new product in a city. It is obvious that people in this city are potential buyers. In order to propagate this product to the public, we need to find several individuals who are the most influential in the network, and hope that, through their propagation, as many people as possible could know this product and then further purchase it.

In our paper, we present a novel algorithm framework, namely, TarIM-SF (Targeted Influence Maximization with Spatial Filtering), to deal with this problem. Given a location-aware social network and a target region, we firstly adopt a spatial filtering model (SFM) to identify the targeted users. Then we will utilize the elegant sampling approach in latest IM solutions [3] to find seed nodes. In order to further

improve the efficiency, we have coarsened the social network. In all, our contributions in this work are as follows:

(i) We relax the target region in LAIM to arbitrary polygons, which is more practical in real applications.

(ii) Secondly, our model can address LAIM from scratch without any assumption for offline processing.

(iii) To the best of our knowledge, we are the first to prove the hardness of LAIM theoretically in detail, and for the large-scale and complex networks, we propose a coarsening-based model that can further improve the efficiency with guaranteed seed quality.

Experiments on real-world datasets Gowalla, Tweets, and Weibo demonstrate that our framework could generate a seed set with theoretically guaranteed quality, which outperforms a series of baseline methods in terms of influence spread quality. Besides, the coarsening-based algorithm can provide superior efficiency.

The rest of paper is organized as follows. Section 2 lists the related studies. Section 3 gives the definition of LAIM with proving its hardness and presents some fundamental knowledge. Afterwards, we discuss the proposed algorithm framework in Section 4. Section 5 shows the theoretical guarantee of seed quality. Section 6 reports the experimental results and some discussion. In Section 7, we conclude the paper.

2. Related Works

Kempe et al. [1] first formulated the influence maximization problem and proved that it is NP-hard in general, but can be approximated with $(1 - 1/e - \epsilon)$ factor. They presented a greedy algorithm with a provable approximation guarantee to solve this problem. However, the greedy algorithm needs to perform the Monte Carlo simulation [7] to obtain the approximate ratio, which has a large time overhead. Furthermore, in order to improve the efficiency and the effect, there has been a large body of research works that can be divided into three types. *Simulation-based* methods accurately estimate influence by simulating the diffusion process repeatedly with a theoretical guarantee. Leskovec et al. [8] proposed a CELF method with the lazy-forward heuristic, which is originally designed to optimize submodular functions in [9], as well as [10–14]. *Heuristic-based* methods are developed to avoid using Monte Carlo simulation at the expense of solution quality. For example, Chen et al. [2] proposed to use local directed acyclic graphs to approximate the influence regions of nodes, while [15] restricting the spread of influence into communities and [6] approximating the influence spread using linear systems. *Sketch-based* methods resolved the inefficiency of Monte Carlo simulations without loss of accurate guarantees. Borges et al. [16] presented a nearly optimal time algorithm for IM under IC model. This method relies on *reverse simulations* of the diffusion process and builds sketches to estimate the influence function efficiently. In subsequential works, techniques for bounding the sketches' size are developed [3, 17–21] and [3, 18, 19] are the representative ones that exhibit higher efficiency in all sketch-based methods. Moreover, Liu et al. [22] construct a community-level influence analysis model, instead of focusing on individual-influence, while [23] defining the outer influence of a community and aiming to find the most influential communities, as well as [24] constructing an influential propagation model considering the temporal-interaction between users in the social network.

Recently, more additional demands for IM problem emerged, such as considering the interests of users [25, 26], geographical factor, or some factors of time. Especially in order to meet the location-aware requirements in IM, Li et al. [4] proposed a method to solve location-aware IM, which works by seeking k users to maximize the expected influence spread of the query region. Wang et al. [27] considered the distance between two users and defined the distance-aware IM problem. They proposed a priority based algorithm with $(1-1/e)$-approximation ratio. The authors in [28] also studied the DAIM problem, considering the distance between the locations and the users. Zhu et al. [29] proposed Gaussian based and distance based mobility models, to derive the location-aware propagation probability in LBSN. Zhou et al. [30] take users' historical mobility behaviour into account and study the IM problem under O2O model. Li et al. [31] aim to find several seed users to maximize geographic spanning regions (MGSR) in the query region, while Li et al. [32] assume that users have their location preference and solve the IM problem for the targeted users. Furthermore, some works focus on spatial-temporal IM problem [33, 34], which aims to find k best trajectories to be attached with an advertisement and maximizes the number of influenced users. Besides the location, the interests/topics of users are also taken into consideration in IM, and [35] proposed an algorithm that returns top-k topics related to the query of a user. Su et al. [36] take not only users' interests but also their preference for locations into account, to find the targeted users, and then seek seeds to maximize the influence for targeted users.

3. Problem Statements and Preliminaries

3.1. Problem Definition

Definition 1 (LAIM). Given a location-aware social network $G = (V, E)$ where each node $v \in V$ is associated with a location (denoted as $v.\phi$), a budget k, and a target region Q, the location-aware influence maximization (LAIM) aims to find k seed nodes (denoted as S) from V, such that the influence spread from S can reach the most number of nodes in Q.

We show the hardness of LAIM problem under *Independence Cascade* (IC) model, which is one of the most popular diffusion models [1]. Before that, we first define a problem called Subset Cover, which will be utilized in the following content.

Definition 2 (subset cover). There are an element set $U = \{u_1, u_2, \ldots, u_n\}$, a subset $U_q = \{u_{q1}, u_{q2}, \ldots, u_{qt}\}$ of U, and a collection of subsets $S = \{S_1, S_2, \ldots, S_m\}$ of U, and we wish to know whether there exist k subsets in S, whose union is equal to U_q.

Theorem 3. *The location-aware influence maximization problem is NP-hard for IC model.*

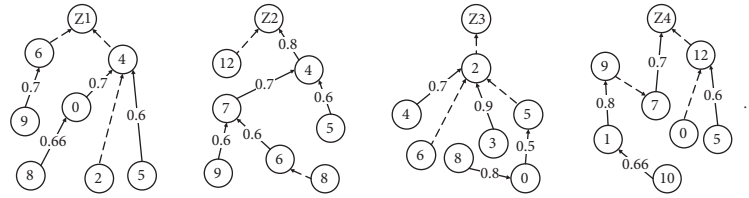


Figure 1: Several hypergraphs constructed from g' (dotted line represents there is a path from a node to another and the path passes through other nodes; real line represents a directed edge).

Proof. From [1], we know that the influence maximization is NP-hard by reduction from Set Cover. Here we can prove that Subset Cover problem above is also NP-hard by reduction from Set Cover problem; the process is as follows.

For S and U in Set Cover problem, we get a subset $U_q = \{u_{q1}, u_{q2}, \ldots, u_{qt}\}$ from U, and we get a new set $U' = \{u_1, u_2, \ldots, u_n, u_{n+1}, u_{n+2}, \ldots, u_{n+t}\}$ by adding t different elements to U, while getting a new collection $S' = \{S'_1, S'_2, \ldots, S'_m\}$ of V corresponding S, where $S'_i = S_i \cup \{u_{n+j} \mid \text{if } u_{qj} \in S_i,\ \forall j \in [t]\}$. This process can be completed in polynomial time. When we find a solution A for Set Cover problem, the corresponding subsets A' in S' can cover all nodes in U'; and if we find the solution of Subset Cover problem, Set Cover problem can also be solved. Based on this, we are able to construct a corresponding directed graph with $(m + t)$ nodes like the proof in [1]: there are node i corresponding to each S_i in S, node j corresponding to each element u_{qj} in U_q, and a directed edge (i, j) with activity p_{ij}, where $p_{ij} = 1$ if $u_{qj} \in S_i$; otherwise it is equal to 0. The Subset Cover problem is equivalent to deciding if there is a set A in this graph with $\sigma_A \geq (t + k)$, where σ_A denotes the influence spread of node set A. Initially, if we find a set A which makes $\sigma_A \geq (t + k)$, the Subset Cover problem will be solved, and if all k nodes corresponding to sets in solution of Subset Cover are activated, all t nodes corresponding to set U_q will be activated. □

3.2. Sampling Technique. Borgs et al. [16] introduced a sampling method called RIS, which first constructs a suitable number of sketches from different target nodes reversing DFS (referred to as RR sets $\mathscr{R}$) and then finds out k users as the seed nodes with the maximum coverage of $\mathscr{R}$. The process of constructing RR sets is as follows.

Firstly, given an edge-weighted graph $G = (V, E)$, we denote $g = (V, E, p)$ as the influence graph of G, where p is the propagation probability for edges between two user nodes. We delete every edge e in G with probability $(1 - p_e)$. After that, we need to randomly choose one node in V and then construct a hypergraph and get the RR set for it.

Definition 4 (RR set). Let v be a node in V. A RR set for v is generated by firstly sampling a graph g' from g and then taking the set of nodes in g' that can reach v.

For instance, from hypergraphs in Figure 1, we can get the RR sets as follows: $R_1 = \{Z1, 6, 4, 0, 2, 5, 9, 8, \ldots\}$, $R_2 = \{Z2, 12, 4, 7, 5, 9, 6, 8, \ldots\}$, $R_3 = \{Z3, 2, 6, 4, 3, 5, 0, 8, \ldots\}$, $R_4 = \{Z4, 7, 12, 9, 0, 5, 1, 10, \ldots\} \cdots$. Through the construction of $\mathscr{R}$, we can seek k nodes as seeds which have the maximum coverage of $\mathscr{R}$.

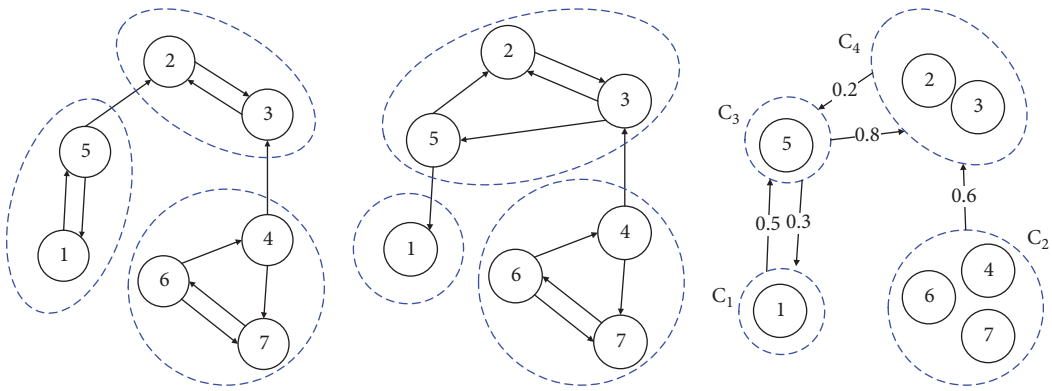
3.3. Coarsening Method

Definition 5 ([37] coarsened influence graph). Given a social network $G = (V, E)$, let $g = (V, E, p)$ be an influence graph and $P = \{C_j\}_{j\in[\tau]}$ be a partition of V, where each C_j is strongly connected (SC). Then, a coarsened influence graph obtained from g is defined as a vertex-weighted influence graph $H = (W, F, q, w)$, where

$$\begin{aligned} W &= \left\{C_j \mid j \in [\tau]\right\}, \\ F &= \left\{\left(C_x, C_y\right) \mid C_x \neq C_y,\ \exists (u, v) \in E,\ u \in C_x,\ v \in C_y\right\}, \\ q_{C_xC_y} &= 1 - \prod_{\substack{(u,v)\in E \\ u\in C_x, v\in C_y}} \left(1 - p_{uv}\right), \quad \forall \left(C_x, C_y\right) \in F, \\ w\left(C_j\right) &= \left|C_j\right|, \quad \forall C_j \in W \end{aligned} \tag{1}$$

The mapping $V \longrightarrow W$ is defined as $\pi(v) = C_j$ such that $v \in C_j$. For a vertex set S, we let $\pi(S) = \{\pi(v) \mid v \in S\}$ and $F = \{(\pi(u), \pi(v)) \mid \pi(u) \neq \pi(v), (u, v) \in E)\}$. We follow the same coarsening process as [37]. Given an influence graph g, whose distribution is D_g, we first construct subgraphs $G_1 = (V_1, E_1), \ldots G_r = (V_r, E_r)$, which are random graphs sampled from D_g. In these subgraphs, we identify the vertex sets connected to each other in all subgraphs, so that we get a partition $P = \{C_1, C_2, \ldots, C_\tau\}$ of g and corresponding $H = (W, F, q, w)$ (see Figure 2).

3.4. Spatial Filtering Model. In order to figure out which nodes fall into region Q, the most intuitive way is to compare the location of each node with the boundary of Q, which is costly when Q is complex or $|V|$ is very large. Herein, we will adopt an efficient method for this task. Our method works by comparing the convex hull of nodes with Q and iteratively removing those nodes falling out of Q. Finally, we can end with a group of nodes whose convex hull is inside Q. In this manner, we avoid enumerating all nodes in V. Notably, for a point set T, we could use *Graham Scan* method [38] to seek its convex hull. Let $T = \{v_1.\phi, \ldots, v_n.\phi\}$ be the nodes' locations and $(q_1, \ldots, q_{m_q})$ be the boundary of Q as a point sequence. Then the process of finding target nodes in Q is shown in Algorithm 1.



(a) Subgraph 1 (b) Subgraph 2 (c) Coarsened influence graph

Figure 2: Two subgraphs of g and the coarsened influence graph H.

Input:
A point set $T = \{v_1.\phi, \ldots, v_n.\phi\}$, a polygon $Q = (q_1, \ldots, q_{m_q})$;
Output:
The points of T within Q.
1: Initialize $i = 1, T_i = T, \mathscr{C}'_i = \emptyset, \mathscr{C}'_0 = \emptyset$;
2: **repeat**
3: Compute the convex hull $\mathscr{C}_i$ of T_i and compare $\mathscr{C}_i$ with Q;
4: **if** $((q_1, \ldots, q_{m_q})$ are all outside $\mathscr{C}_i) \bigwedge (\forall c \in \mathscr{C}_i$ are in $Q)$ **then**
5: **return** $\{\mathscr{C}'_1, \mathscr{C}'_2, \ldots, \mathscr{C}'_{i-1}, T_i\}$.
6: **else if** $((q_1, \ldots, q_{m_q})$ are all outside $\mathscr{C}_i) \bigwedge (\forall c \in \mathscr{C}_i$ are outside $Q) \bigwedge (\mathscr{C}_i \cap Q = \emptyset)$ **then**
7: **return** $\emptyset$.
8: **else then**
9: continue;
10: **end if**
11: **end if**
12: $\mathscr{C}'_i = \{v_{i1}.\phi, \ldots, v_{ij}.\phi\}$ is in Q, $1 \le j \le i$;
13: $T_{i+1} = T_i - \mathscr{C}_i$, $i = i + 1$;
14: **until** $T_i = \emptyset$.
15: **return** $\{\mathscr{C}'_1, \mathscr{C}'_2, \ldots, \mathscr{C}'_{i-1}\}$.

Algorithm 1: SFM (T, Q).

4. TarIM-SF Framework

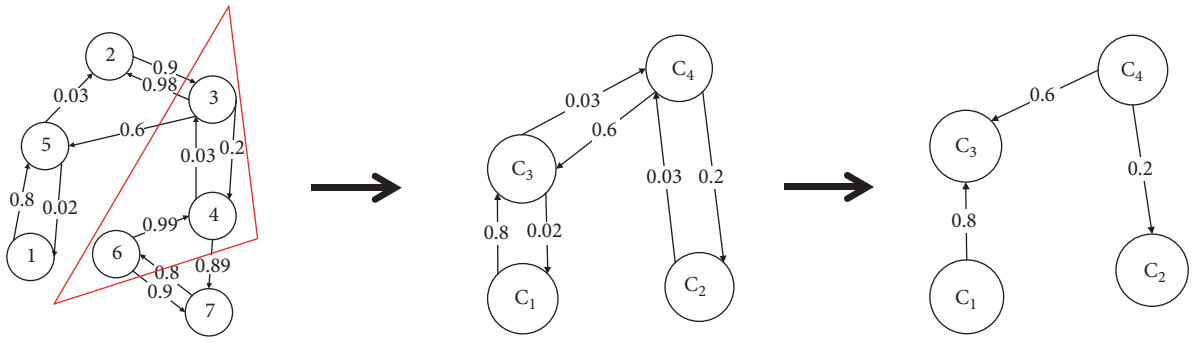
Here we describe our TarIM-SF framework in detail. In our LAIM problem, target users change from the whole network in classic IM problem into users in a target region. As the construction of RR sets starts from the target users, it is reasonable for us to construct enough RR sets $\mathscr{R}$ over the whole network for users in the target region within our problem and then choose the node set s which maximizes the coverage of $\mathscr{R}$ (referred to as $F_{\mathscr{R}}(s)$). Based on this idea, we first need to identify the users in the target region before constructing RR sets, which is addressed using the method proposed in Section 3.4. Moreover, in order to improve the efficiency of constructing RR sets, we have coarsened the network using the method proposed in Section 3.3. More details are given in Algorithm 2.

For instance, in Figure 3, given a location-aware social network G, $k = 1$, and a query region Q (such as a triangle), we first use SFM (line 3) to identify the goal users $\{3, 4, 6\}$, then we coarsen the whole network, and we get the partition $P = \{C_1, C_2, C_3, C_4\}$, where $C_1 = \{1\}, C_2 = \{4, 6, 7\}, C_3 = \{5\}, C_4 = \{2, 3\}$ (line 4). Next we will sample the coarsened influence graph and construct enough RR sets for the goal users (line 5). In the coarsened influence graph, the probability of partition node C_4 being chosen to construct RR set is 1/3, and there is 2/3-probability for node C_2. As a result, we get RR sets: $R_1 = \{C_2, C_4\}, R_2 = \{C_4\}, R_3 = \{C_2, C_4\}$. We can see that node C_4 has the maximum coverage of $\mathscr{R}$, including R_1, R_2, R_3. In turn for the original network graph, we choose one user node randomly in C_4 as the seed, such as node 2 (lines 6-10).

In our framework, the first step is to identify the users in the target region Q, whose complexity depends on the location distribution of all nodes. In case that all users' locations are uniformly randomly distributed, the time complexity for identifying users within the target region is $O(n^{1/3}m_q)$

Input:
A location-aware social network $G = (V, E)$, a query polygon Q, and a budget k;
Output:
A seed set S.
1: Initialize $S = \emptyset$, a set of goal users $qs = \emptyset$;
2: $T \longleftarrow$ location points of V;
3: $qs \longleftarrow$ SFM (T, Q);
4: $G' \longleftarrow$ Coarsening (G), according to Algorithm 1 in [37];
5: $\mathscr{R} \longleftarrow$ Sampling (G', qs, k, ϵ), according to Algorithm 2 in [3];
6: **for** $i = 0$ to $k - 1$ **do**
7: $s' \longleftarrow$ seek a node in G' with the maximum marginal coverage for $\mathscr{R}$;
8: $s \longleftarrow$ choose a vertex in s' randomly;
9: $S \longleftarrow S \cup \{s\}$;
10: **end for**
11: **return** S.

Algorithm 2: TarIM-SF (G, Q, k, ϵ).



Step 1: A query Step 2: Coarsening Step 3: Sampling

Figure 3: An example.

under our spatial filtering model, where $n = |V|$ and m_q is the number of points for Q. The second step is to coarsen the network, which requires $O(r(|V| + |E|))$ time, where r denotes the number of random subgraphs sampled from g. Afterwards, we use algorithm in [3] to seek solution in coarsened influence network, and the time of this step is spent on constructing RR sets for target region. The complexity of this process is $O(\sum_{j\in[\mathscr{R}]} w(R_j))$, $w(R_j)$ denoting the edges number of the i-th RR set, where $|\mathscr{R}|$ is decided by the parameters λ' and λ^* in [3], as well as the number of nodes and edges in the coarsened influence graph.

5. Effectiveness Study

We will subsequently conduct a theoretical study over the seed quality of our algorithm framework for both cases when coarsening is present or not.

5.1. Seed Quality without Coarsening. In [1], it has been proved that, under Independent Cascade model, the result influence function $\sigma(\cdot)$ is sunmodular. Here we define $\sigma(\cdot)$ as the target influence spread, which is the number of influenced nodes in target region. It is easy to see that, for any sets S and T, $S \subseteq T$, and any elements v, $\sigma(S \cup \{v\}) - \sigma(S) \geq \sigma(T \cup \{v\}) - \sigma(T) \geq$ also holds. Hence, the function $\sigma(\cdot)$ is submodular.

Theorem 6 (see [1]). *For a nonnegative, monotone submodular function f, let S_k^o be a size-k set by selecting one element at a time, each time choosing the element which has the maximum marginal function value. Assuming S_k^* is the set that maximizes the value of f over all k-element sets, then $f(S_k^o) \geq (1 - 1/e) \cdot f(S_k^*)$; in other words, S_k^o guarantees a $(1-1/e)$-approximation.*

So if we get a seed set S_{out} by adopting a greedy algorithm, S_{out} is a $(1 - 1/e)$-approximate solution for the location-aware influence maximization problem. Then we will show the performance guarantee when we adopt IMM method [3] as the greedy approach.

Lemma 7 (see [16]). *For any seed set S and any vertex v, the probability that a diffusion process from S can activate v equals the probability that S overlaps an RR set for v.*

We generate a sizeable set $\mathscr{R}$ of random RR sets for the nodes in target region Q, and for any seed set S, the fraction $F_{\mathscr{R}}(S)$ of RR sets in $\mathscr{R}$ covered by S is the unbiased estimator

of $\mathbb{E}(\sigma(S))/n_q$, where n_q is the number of vertices in target region Q. In TIM$^+$ [17], it has been proved that the solution which covers the maximum number of RR sets provides a $(1-1/e-\epsilon)$-approximation with at least $(1-1/n^\ell)$ probability, but the number of RR sets is at least λ/OPT, where OPT is the maximum expected influence of any size-k nodes set in G and λ is a function of k, ℓ, n, and ϵ. In IMM, it seeks a tighter lower bound LB of OPT than TIM$^+$. Next, based on the analysis of the performance guarantee in the IMM, we will describe the parameter settings and performance guarantee in our framework.

Let $R_1, R_2, \ldots, R_\theta$ be the sequence of generated RR sets for nodes in the target region Q. Let S_k be any size-k seed set in G and x_i be random variable that equals 0 if $S_k \cap R_i = \emptyset$ and 1 otherwise; then based on Lemma 7, we have

$$\mathbb{E}\left[\sigma\left(S_k\right)\right] = \frac{n_q}{\theta} \cdot \mathbb{E}\left[\sum_{i=1}^{\theta} x_i\right]. \tag{2}$$

Consider S_k^* is the size-k node set with the maximum expected influence; let $OPT = \mathbb{E}[\sigma(S_k^*)]$. From (2), we can get that $n_q \cdot F_{\mathscr{R}}(S_k^*)$ is an unbiased estimator of OPT. By Corollary 2 in which we set $p = OPT/n_q$ and Lemma 3 in [3], we have the following.

Lemma 8. *Let $\epsilon_1 > 0$, $\delta_1 \in (0,1)$, and*

$$\theta_1 = \frac{2n_q \cdot \log\left(1/\delta_1\right)}{OPT \cdot \epsilon_1^2}, \tag{3}$$

if $\theta \geq \theta_1$, then $n_q \cdot F_{\mathscr{R}}(S_k^*) \geq (1-\epsilon_1) \cdot OPT$ holds with at least $(1-\delta_1)$ probability.

Assuming that our framework without coarsening returns a solution S_k', and if $n_q \cdot F_{\mathscr{R}}(S_k^*) \geq (1-\epsilon_1) \cdot OPT$ holds, according to the properties of the greedy approach, then

$$\begin{aligned} n_q \cdot F_{\mathscr{R}}\left(S_k'\right) &\geq \left(1-\frac{1}{e}\right) \cdot n_q \cdot F_{\mathscr{R}}\left(S_k^*\right) \\ &\geq \left(1-\frac{1}{e}\right) \cdot \left(1-\epsilon_1\right) \cdot OPT \end{aligned} \tag{4}$$

Lemma 9. *Let $\epsilon_1 < \epsilon$, $\delta_2 \in (0,1)$, and*

$$\theta_2 = \frac{(2-2/e) \cdot n_q \cdot \log\left(\binom{n}{k}/\delta_2\right)}{OPT \cdot \left(\epsilon - (1-1/e)\cdot\epsilon_1\right)^2}, \tag{5}$$

if (4) holds and $\theta \geq \theta_2$, then $\mathbb{E}(\sigma(S_k')) \geq (1-1/e-\epsilon) \cdot OPT$ holds with at least $(1-\delta_2)$ probability.

Based on Lemmas 8 and 9, we have the following.

Theorem 10. *Given any $\epsilon_1 \leq \epsilon$ and any $\delta_1, \delta_2 \in (0,1)$ with $\delta_1 + \delta_2 \leq 1/n^\ell$, if $\theta \geq \max\{\theta_1, \theta_2\}$, the result which IMM returns is a $(1-1/e-\epsilon)$-approximate solution with $(1-1/n^\ell)$ probability.*

For the parameters in Theorem 10, we set $\delta_1 = \delta_2 = 1/(2n^\ell)$, and under this setting, θ is minimized when $\theta_1 = \theta_2$, and $\epsilon_1 = \epsilon \cdot \alpha/((1-1/e)\cdot\alpha\cdot\beta)$, where

$$\begin{aligned} \alpha &= \sqrt{\ell \log n + \log 2}, \\ \beta &= \sqrt{\left(1-\frac{1}{e}\right)\cdot\left(\log\binom{n}{k} + \ell\log n + \log 2\right)}. \end{aligned} \tag{6}$$

In this case, $\theta = (2n_q \cdot ((1-1/e)\cdot\alpha+\beta)^2)/(OPT\cdot\epsilon^2)$, and conversely if we set $\theta = \lambda^*/OPT$, where

$$\lambda^* = \frac{2n_q \cdot \left((1-1/e)\cdot\alpha+\beta\right)^2}{\epsilon^2}, \tag{7}$$

the seed set S_k' that covers the maximum number of RR sets is a $(1-1/e-\epsilon)$-approximation. However, as OPT is unknown in advance, we will find a tight lower bound LB of OPT as IMM method. In the sampling phase in IMM, Lemma 6, Lemma 7, and Lemma 8 in [3] proved that $LB \leq OPT$ and LB is close to OPT. And based on that, we can also prove that $LB \leq OPT$ and LB is a tight lower bound of OPT with a high probability by changing n to n_q.

Theorem 11. *With at least $(1-1/n^\ell)$ probability, sampling algorithm in our framework returns a set $\mathscr{R}$ of RR sets with $|\mathscr{R}| \geq \lambda^*/OPT$, where λ^* is as defined in (7).*

Combining Theorem 10 and Theorem 11, our algorithm framework without coarsening can get a solution S_{out} which is a $(1-1/e-\epsilon)$-approximation with a high probability.

5.2. Seed Quality for Coarsening Method. In this part, we will show the result seed set S_{out} can achieve $(1-1/e-\epsilon)\cdot\alpha_g$-approximation, when coarsening technique is adopted to improve the efficiency. Based on the study in [37], for the target region Q, we will get the following equation:

$$\inf_g(Q,S) = \sum_{E'\subseteq E} p_{E'|E} \cdot R_{(V,E')}(Q,S), \tag{8}$$

where $p_{E'|E} = \prod_{e\in E'} p_e \cdot \prod_{e\in E\setminus E'}(1-p_e)$, $R_g(Q,S)$ is the number (sum of weights) of vertices in Q that are reachable from S in g, and $\inf_g(Q,S)$ is the number of vertices in Q that S can activate in g, which equals $\sigma(S)$ in Section 5.1 and is submodular.

In the coarsened influence graph $H = (W, F, q, w)$, we also have

$$\inf_H(Q, \pi(S)) = \sum_{Y\subseteq F} q_{Y|F} \cdot R_{(W,Y)}(Q, \pi(S)). \tag{9}$$

It denotes the sum of weights for nodes in Q that $\pi(S)$ can activate in H. After coarsening the network, we define $I = (V, E, p')$, where $p'_{uv} = 1$ if $u, v \in C_j$ for all $j \in [\tau]$; otherwise $p'_{uv} = p_{uv}$, and $\inf_I(Q,S)$ is the number of users in target region Q that S can activate in I.

Here, we show the relationship between H and g in terms of influence function through I, which has the same structure as g.

Lemma 12. *For any Q and any $S \subseteq V$, $\inf_I(Q,S) = \inf_H(Q,\pi(S))$.*

Proof. For any u and v in V, "u can reach v through the edges in E with p" if and only if "$\pi(u)$ can reach $\pi(v)$ through the edges in F with q", since every subgraph (C_j, E_j), $j \subseteq [\tau]$, is strongly connected. Therefore, it holds that $R_{(V,E)}(Q,S) = R_{(W,Y)}(Q,\pi(S))$. Thus,

$$\begin{aligned}\inf_I(Q,S) &= \sum_{E'\subseteq E} p'_{E'|E}\cdot R_{(V,E')}(Q,S)\\ &= \sum_{Y\subseteq F} q_{Y|F}\cdot R_{(W,Y)}(Q,\pi(S))\\ &= \inf_H(Q,\pi(S))\end{aligned} \tag{10}$$

□

For g and I, we also can find the relationship between them as follows.

Lemma 13. *For any Q and any $S \subseteq V$, $\inf_g(Q,S) \le \inf_I(Q,S)$.*

Proof. g and I are the influence graph with the same structure, but $p_e \le p'_e$ for every edge e. So $\inf_g(Q,S) \le \inf_I(Q,S)$ for any $S \subseteq V$.

For any subgraph $g[V'] = (V', E')$ of g, $V' \subseteq V$, its *strongly connected reliability*, denoted as $Rel(g[V'])$, is defined the same as in Equation 14 in [37] and indicates the probability that $g[V']$ is strongly connected. □

Lemma 14. *$\inf_I(Q,S) \le \prod_{j\in[\tau]} Rel(g[C_j])^{-1}\cdot \inf_g(Q,S)$, for any Q and $S \subseteq V$.*

Proof. For each $j \in [\tau]$, we define $E_j = \{(u,v) \in E \mid u, v \in C_j\}$ and $E_0 = E \setminus (\bigcup_{j\in[\tau]} E_j)$. For $X_j \subseteq E_j (j \in [\tau])$ and $X_0 \subseteq E_0$, $R_{(V,X_0\cup\bigcup_j X_j)}(S) = R_{(V,X_0\cup\bigcup_j E_j)}(S)$ holds if every subgraph (C_j, X_j) for $j \in [\tau]$ is strongly connected. Thus

$$\begin{aligned}\inf_g(Q,S) &= \sum_{X_1\subseteq E_1} p_{X_1|E_1}\cdots\sum_{X_\tau\subseteq E_\tau} p_{X_\tau|E_\tau}\\ &\quad\sum_{X_0\subseteq E_0} p_{X_0|E_0}\cdot R_{(V,X_0\cup\bigcup_j X_j)}(S)\\ &\ge \sum_{\substack{X_1\subseteq E_1\\ (C_1,X_1)\text{ is SC}}} p_{X_1|E_1}\cdots\sum_{\substack{X_\tau\subseteq E_\tau\\ (C_\tau,X_\tau)\text{ is SC}}} p_{X_\tau|E_\tau}\\ &\quad\sum_{X_0\subseteq E_0} p_{X_0|E_0}\cdot R_{(V,X_0\cup\bigcup_j X_j)}(S)\\ &= \prod_{j\in[\tau]} Rel\left(g\left[C_j\right]\right)\cdot\inf_I(Q,S)\end{aligned} \tag{11}$$

□

Theorem 15. *For any Q and any $S \subseteq V$, $\inf_g(Q,S) \le \inf_H(Q,\pi(S)) \le \prod_{j\in[\tau]} Rel(g[C_j])^{-1}\cdot\inf_g(Q,S)$.*

Let S^* and T^* be the optimal solutions of size k for g and H, respectively. Based on Lemma 12 and Lemma 13, we can sure that $\inf_H(Q,T^*) \ge \inf_g(Q,S^*)$, and have

$$\begin{aligned}\inf_H(Q,S_{out}) &\ge \left(1-\frac{1}{e}-\epsilon\right)\cdot\inf_H(Q,T^*)\\ &\ge \left(1-\frac{1}{e}-\epsilon\right)\cdot\inf_g(Q,S^*)\end{aligned} \tag{12}$$

Then applying Lemma 14, we have

$$\begin{aligned}&\inf_H(Q,S_{out})\\ &\ge \prod_{j\in[\tau]} Rel\left(g\left[C_j\right]\right)\cdot\left(1-\frac{1}{e}-\epsilon\right)\cdot\inf_g(Q,S^*)\end{aligned} \tag{13}$$

Therefore, our coarsening-based algorithm achieves a $(1-1/e-\epsilon)\cdot\alpha_g$-approximate solution for g, where α_g refers to $\prod_{j\in[\tau]} Rel(g[C_j])$.

6. Results and Discussion

In this section, we conduct experiments on several real-world datasets to test the performance of the proposed algorithm framework. All algorithms are implemented in C++ and run on Ubuntu 16.10 machine with Intel Core i5-6500 quad-core, 3.20GHz, 16GB RAM.

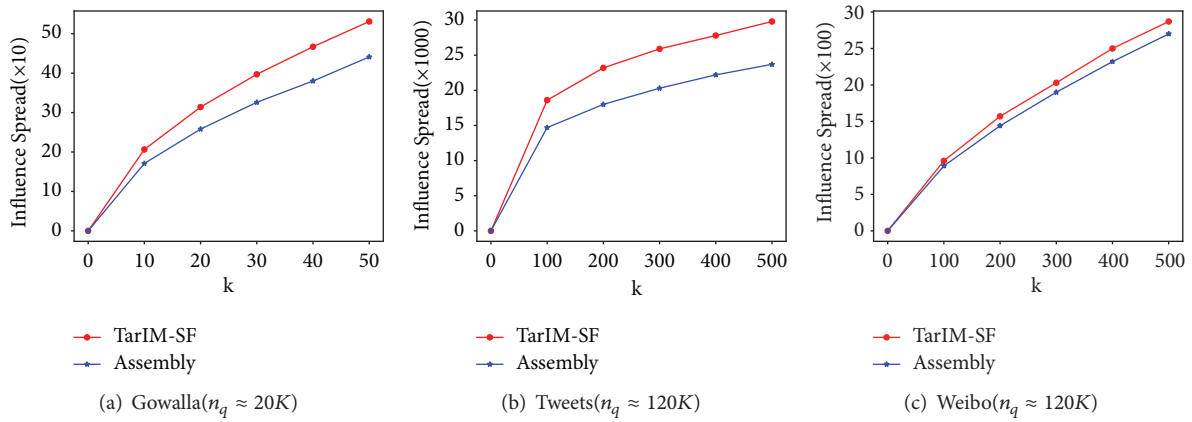
In the following experiments, we use three location-aware social networks, namely, Gowalla, Tweets, and Weibo. The statistics for the datasets are listed in Table 1 (n represents the number of vertices and m represents the number of edges). By default, we use a randomly selected Q for all datasets, and the number of user nodes falling in Q is denoted as n_q. We conduct our experiments on WC model, which is widely used for information diffusion. For the weight of every edge, we set the probability of an edge (u, v) as $1/N_v$, where N_v denotes the in-degree of user node v. In TarIM-SF, we set $\epsilon = 0.1$.

6.1. Comparison with Baseline. We evaluate the performance of our algorithm framework **TarIM-SF** compared with method **Assembly** in [4] under WC model. In order to estimate the performance in general, we selected three regions with fixed n_q for each dataset ($n_q \approx 20K$ for Gowalla, $n_q \approx 120K$ for Tweets, and $n_q \approx 120K$ for Weibo) and reported the average performance, while varying k for Gowalla from 10 to 50 and k for Tweets and Weibo from 100 to 500. In Figure 4, we can see that the target influence spread of seeds in our framework is obviously superior to that of **Assembly**, especially on Tweets and Gowalla.

6.2. Effect of Coarsening. As mentioned above, we adopt coarsening technique so as to improve the efficiency for large-scale social networks. In this part, we did a series of experiments on Weibo, a large-scale network with about a million users. In order to justify how the parameter r in coarsening method will affect the target influence spread of seeds, we report the *Relative Error*, which measures the gap between the real influence spread of seed set we get and its

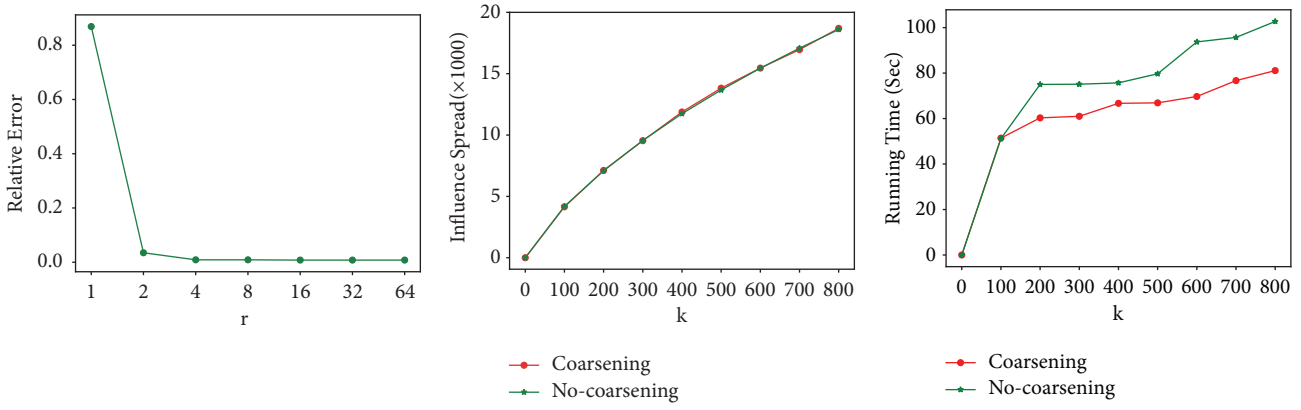
Table 1: Datasets statistic.

Datasets	n	m
Gowalla	196,585	351,452
Tweets	554,372	2,402,720
Weibo	1,020,730	16,490,916



(a) Gowalla($n_q \approx 20K$) (b) Tweets($n_q \approx 120K$) (c) Weibo($n_q \approx 120K$)

Figure 4: Influence spread under WC model.



(a) Effect of r (b) Influence spread($r = 16$) (c) Running time($r = 16$)

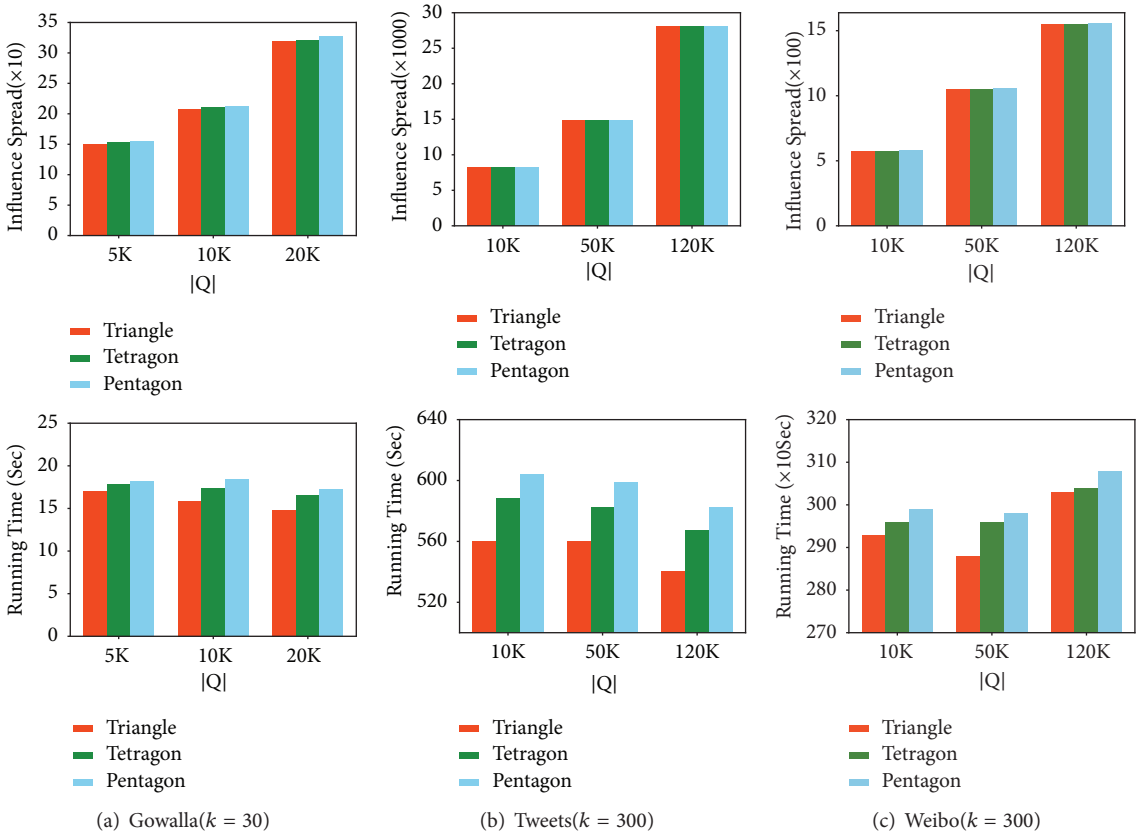
Figure 5: Effect when using coarsening.

estimated influence spread. We set $k = 300$ and the target region as the whole network. Figure 5(a) shows the relative error is decreasing when r becomes bigger. When $r = 16$, the estimated influence spread of seeds using our algorithm with coarsening is nearly equal to the eventual influence spread of seeds without coarsening. Figures 5(b) and 5(c) indicate that the running time for coarsening approach is significantly less than that of noncoarsening one, without loss of seed quality.

6.3. Varying the Size and Shape of Q. We also conducted another group of experiments by varying Q in terms of both size and shape. Specifically, we vary Q as triangle, tetragon, and pentagon, respectively. For each shape, we vary the size at several different levels and report the performance of our algorithm (shown in Figure 6, $|Q| = n_q$). It justifies that our algorithm can work on target region with arbitrary polygon shapes. Besides, the time spent on the first phase in our algorithm framework increases slightly as the shape of Q varied from pentagon to triangle at the same scale, because nodes in convex hull need to be compared with the polygon queried and the fewer the number of polygon edges, the fewer the comparison times and the less time.

7. Conclusions

In this study, we present a novel model that can address LAIM with a target region that can be an arbitrary polygon. Our framework uses a spatial filtering model to initially figure out the nodes falling into the target polygon. Afterwards, a coarsening process is conducted over the network. Then, the state-of-the-art sampling algorithm adopted in traditional IM is used to find the solution. We theoretically prove the



(a) Gowalla(k = 30) (b) Tweets(k = 300) (c) Weibo(k = 300)

Figure 6: Influence spread and running time in terms of size and shape of Q.

influence spread guarantee in both noncoarsened and coarsened cases. Empirical study over three real-world datasets demonstrates that our framework outperforms the baseline algorithm in terms of influence spread and is efficient in large-scale networks.

Data Availability

This study is based on the datasets Gowalla, Tweets, and Weibo provided in [4], and they are available at http://dbgroup.cs.tsinghua.edu.cn/ligl/laim/.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The work is supported by the National Natural Science Foundation of China (Grant No. 61672408), Fundamental Research Funds for the Central Universities (No. JB181505), Natural Science Basic Research Plan in Shaanxi Province of China (No. 2018JM6073), and China 111 Project (No. B16037).

References

[1] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03)*, pp. 137–146, New York, NY, USA, August 2003.

[2] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD*, pp. 1029–1038, USA, July 2010.

[3] Y. Tang, Y. Shi, and X. Xiao, "Influence maximization in near-linear time: a martingale approach," in *Proceedings of the ACM SIGMOD International Conference on Management of Data, (SIGMOD '15)*, pp. 1539–1554, Melbourne, Australia, June 2015.

[4] G. Li, S. Chen, J. Feng, K. Tan, and W. Li, "Efficient location-aware influence maximization," in *Proceedings of the the 2014 ACM SIGMOD international conference*, pp. 87–98, Snowbird, Utah, USA, June 2014.

[5] A. Goyal, W. Lu, and L. V. S. Lakshmanan, "SIMPATH: An efficient algorithm for influence maximization under the Linear Threshold model," in *Proceedings of the 11th IEEE International Conference on Data Mining, ICDM 2011*, pp. 211–220, Canada, December 2011.

[6] K. Jung, W. Heo, and W. Chen, "IRIE: Scalable and robust influence maximization in social networks," in *Proceedings of the 12th IEEE International Conference on Data Mining, ICDM 2012*, pp. 918–923, Belgium, December 2012.

[7] D. J. C. MacKay, "Introduction to monte carlo methods," *Learning in Graphical Models*, vol. 30, no. 90, pp. 175–204, 1998.

[8] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. Vanbriesen, and N. Glance, "Cost-effective outbreak detection in networks,"

in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '07)*, pp. 420–429, New York, NY, USA, August 2007.

[9] M. Minoux, "Accelerated greedy algorithms for maximizing submodular set functions," *Optimization Techniques*, pp. 234–243, 1978.

[10] A. Goyal, W. Lu, and L. V. S. Lakshmanan, "CELF++: optimizing the greedy algorithm for influence maximization in social networks," in *Proceedings of the 20th International Conference Companion on World Wide Web, (WWW '11)*, pp. 47-48, Hyderabad, India, April 2011.

[11] N. Ohsaka, T. Akiba, Y. Yoshida, and K. I. Kawarabayashi, "Fast and accurate influence maximization on large networks with pruned monte-carlo simulations," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pp. 138–144, AAAI Press, 2014.

[12] S. Cheng, H. Shen, J. Huang, G. Zhang, and X. Cheng, "StaticGreedy," in *Proceedings of the the 22nd ACM international conference*, pp. 509–518, San Francisco, California, USA, October 2013.

[13] M. Kimura, K. Saito, and R. Nakano, "Extracting influential nodes for information diffusion on a social network," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 20, pp. 1371–1376, AAAI Press, 2007.

[14] X. Liu, M. Li, S. Li, S. Peng, X. Liao, and X. Lu, "IMGPU: GPU-accelerated influence maximization in large-scale social networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 136–145, 2014.

[15] Y. Wang, G. Cong, G. Song, and K. Xie, "Community-based Greedy algorithm for mining top-K influential nodes in mobile social networks," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '10)*, pp. 1039–1048, ACM, July 2010.

[16] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *Proceedings of the ACM-SIAM, ACM-SIAM*, pp. 946–957, 2014.

[17] Y. Tang, X. Xiao, and Y. Shi, "Influence maximization: Near-optimal time complexity meets practical efficiency," in *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, SIGMOD 2014*, pp. 75–86, USA, June 2014.

[18] H. T. Nguyen, M. T. Thai, and T. N. Dinh, "Stop-and-Stare: Optimal sampling algorithms for viral marketing in billion-scale networks," in *Proceedings of the 2016 ACM SIGMOD International Conference on Management of Data, SIGMOD 2016*, pp. 695–710, USA, July 2016.

[19] K. Huang, S. Wang, G. Bevilacqua, X. Xiao, and L. V. S. Lakshmanan, "Revisiting the Stop-and-Stare algorithms for influence maximization," in *Proceedings of the 43rd International Conference on Very Large Data Bases, VLDB 2017*, pp. 913–924, Germany, September 2017.

[20] T. Dinh, H. Nguyen, P. Ghosh, and M. Mayo, "Social influence spectrum with guarantees: computing more in less time," in *Computational Social Networks*, vol. 9197 of *Lecture Notes in Computer Science*, pp. 84–103, Springer International Publishing, Cham, 2015.

[21] H. T. Nguyen, T. N. Dinh, and M. T. Thaip, "Cost-aware Targeted Viral Marketing in billion-scale networks," in *Proceedings of the IEEE INFOCOM 2016 - IEEE Conference on Computer Communications*, pp. 1–9, San Francisco, CA, USA, April 2016.

[22] Y. Liu, D. Pi, and L. Cui, "Mining community-Level influence in microblogging network: A case study on sina weibo," *Complexity*, vol. 2017, 16 pages, 2017.

[23] J. Li, X. Wang, and K. Deng, "Most influential commuity search over large social networks," *ICDE, IEEE*, pp. 871–882, 2017.

[24] N. Alduaiji, A. Datta, and J. Li, "Influence propagation model for clique-based community detection in social networks," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 2, pp. 563–575, 2018.

[25] Y. Li, D. Zhang, and K.-L. Tan, "Real-time targeted influence maximization for online advertisements," in *Proceedings of the Vldb Endowment*, pp. 1070–1081, Republic of Korea, 2015.

[26] S. Chen, J. Fan, G. Li, J. Feng, K.-L. Tan, and J. Tang, "Online topic-aware influence maximization," in *Proceedings of the 41st International Conference on Very Large Data Bases, VLDB 2015*, pp. 666–677, USA, September 2015.

[27] X. Wang, Y. Zhang, W. Zhang, and X. Lin, "Efficient Distance-Aware Influence Maximization in Geo-Social Networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 3, pp. 599–612, 2017.

[28] M. Zhong, Q. Zeng, Y. Zhu, J. Li, and T. Qian, "Sample location selection for efficient distance-aware influence maximization in geo-social networks," in *Database Systems for Advanced Applications*, vol. 10827 of *Lecture Notes in Computer Science*, pp. 355–371, Springer International Publishing, Cham, 2018.

[29] W. Zhu, W. Peng, L. Chen, K. Zheng, and X. Zhou, "Exploiting viral marketing for location promotion in location-based social networks," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 11, no. 2, pp. 1–28, 2016.

[30] T. Zhou, J. Cao, B. Liu, S. Xu, Z. Zhu, and J. Luo, "Location-based influence maximization in social networks," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 1211–1220, New York, NY, USA, 2015.

[31] J. Li, T. Sellis, J. S. Culpepper, Z. He, C. Liu, and J. Wang, "Geo-social influence spanning maximization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 8, pp. 1653–1666, 2017.

[32] X. Li, X. Cheng, S. Su, and C. Sun, "Community-based seeds selection algorithm for location aware influence maximization," *Neurocomputing*, vol. 275, pp. 1601–1613, 2018.

[33] L. Guo, D. Zhang, G. Cong, W. Wu, and K.-L. Tan, "Influence maximization in trajectory databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 3, pp. 627–641, 2017.

[34] D. Zhang, L. Guo, L. Nie, J. Shao, S. Wu, and H. T. Shen, "Targeted advertising in public transportation systems with quantitative evaluation," *ACM Transactions on Information and System Security*, vol. 35, no. 3, pp. 1–29, 2017.

[35] J. Li, C. Liu, J. X. Yu, Y. Chen, T. Sellis, and J. S. Culpepper, "Personalized influential topic search via social network summarization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 7, pp. 1820–1834, 2016.

[36] S. Su, X. Li, X. Cheng, and C. Sun, "Location-aware targeted influence maximization in social networks," *Journal of the Association for Information Science and Technology*, vol. 69, no. 2, pp. 229–241, 2018.

[37] N. Ohsaka, T. Sonobe, S. Fujita, and K.-I. Kawarabayashi, "Coarsening massive influence networks for scalable diffusion analysis," in *Proceedings of the 2017 ACM SIGMOD International Conference on Management of Data, SIGMOD 2017*, pp. 635–650, USA, May 2017.

[38] R. L. Graham, "An efficient algorithm for determining the convex hull of a finite planar set," *Information Processing Letters*, vol. 1, no. 4, pp. 132-133, 1972.