Predicting Quality of Service via Leveraging Location Information

Liang Chen, Fenfang Xie, Zibin Zheng, and Yaoming Wu

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

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

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

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 Location-Aware QoS prediction approach, by exploiting Neural network and Factorization Machine (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.

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. \( Q \in \mathbb{R}^{m \times n} \) 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...
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 collaborative 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 2: A toy example of the influence of the location information.
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: \( (1,0), (0,1,0), (1,0,0,0), (1,1,0,0), (0,1,0,0) \)). 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 techniques. Thirdly, the inner products of embedding vectors (i.e., the user information embedding vectors and the service information embedding vectors (e.g., \( V_1, V_4, V_6, V_{12}, V_{15}, V_{18} \)), along with the weighted sum of feature vectors (e.g., \( w_1x_1, w_4x_4, w_6x_6, w_{12}x_{12}, w_{15}x_{15}, w_{18}x_{18} \)), are leveraged to predict the QoS values. Finally, the prediction score for pairwise user-service can be acquired (e.g., \( \hat{y} \)).

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.

<table>
<thead>
<tr>
<th>Web Service ID</th>
<th>WSDL Address</th>
<th>Provider Name</th>
<th>Country Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td><a href="http://www.law.uni-sofia.bg/_vti_bin/People.asmx?wsdl">http://www.law.uni-sofia.bg/_vti_bin/People.asmx?wsdl</a></td>
<td>uni-sofia.bg</td>
<td>Bulgaria</td>
</tr>
<tr>
<td>3</td>
<td><a href="http://www.etfo.ca/_vti_bin/Authentication.asmx?wsdl">http://www.etfo.ca/_vti_bin/Authentication.asmx?wsdl</a></td>
<td>etfo.ca</td>
<td>Canada</td>
</tr>
<tr>
<td>5</td>
<td><a href="http://www.emris.cz/_vti_bin/BusinessDataCatalog.asmx?wsdl">http://www.emris.cz/_vti_bin/BusinessDataCatalog.asmx?wsdl</a></td>
<td>emris.cz</td>
<td>Czech Republic</td>
</tr>
</tbody>
</table>

Table 2: Original location information of users.

<table>
<thead>
<tr>
<th>User ID</th>
<th>IP Address of the User</th>
<th>Country</th>
<th>Longitude</th>
<th>Latitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12.108.127.138</td>
<td>America</td>
<td>40.44</td>
<td>79.96</td>
</tr>
<tr>
<td>2</td>
<td>122.1.115.91</td>
<td>Japan</td>
<td>36</td>
<td>138</td>
</tr>
<tr>
<td>3</td>
<td>128.233.252.11</td>
<td>Canada</td>
<td>52.13</td>
<td>106.67</td>
</tr>
<tr>
<td>4</td>
<td>129.242.19.196</td>
<td>Norway</td>
<td>69.67</td>
<td>18.97</td>
</tr>
<tr>
<td>5</td>
<td>202.38.99.68</td>
<td>China</td>
<td>39.93</td>
<td>116.39</td>
</tr>
</tbody>
</table>

Table 3: Processed location information of Web services.

<table>
<thead>
<tr>
<th>Web Service ID</th>
<th>Country Name</th>
<th>ASN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>United States</td>
<td>33070</td>
</tr>
<tr>
<td>2</td>
<td>Bulgaria</td>
<td>5421</td>
</tr>
<tr>
<td>3</td>
<td>Canada</td>
<td>36031</td>
</tr>
<tr>
<td>4</td>
<td>China</td>
<td>23650</td>
</tr>
<tr>
<td>5</td>
<td>Czech Republic</td>
<td>43541</td>
</tr>
</tbody>
</table>

Table 4: Processed location information of users.

<table>
<thead>
<tr>
<th>User ID</th>
<th>Country Name</th>
<th>ASN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>America</td>
<td>7018</td>
</tr>
<tr>
<td>2</td>
<td>Japan</td>
<td>4713</td>
</tr>
<tr>
<td>3</td>
<td>Canada</td>
<td>22950</td>
</tr>
<tr>
<td>4</td>
<td>Norway</td>
<td>224</td>
</tr>
<tr>
<td>5</td>
<td>China</td>
<td>4538</td>
</tr>
</tbody>
</table>

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.

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
The one-hot 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

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

wherein $f$ is the length of the feature vector. $V_i$ is the $i$th 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 $i$th 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.

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

<table>
<thead>
<tr>
<th>Name</th>
<th>One-hot Encoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asia</td>
<td>1000000</td>
</tr>
<tr>
<td>Africa</td>
<td>0100000</td>
</tr>
<tr>
<td>North America</td>
<td>0010000</td>
</tr>
<tr>
<td>South America</td>
<td>0001000</td>
</tr>
<tr>
<td>Antarctica</td>
<td>0000100</td>
</tr>
<tr>
<td>Europe</td>
<td>0000010</td>
</tr>
<tr>
<td>Oceania</td>
<td>0000001</td>
</tr>
</tbody>
</table>
The formula of cross features is as follows:

\[ \langle V_i, V_j \rangle = \sum_{l=1}^{k} v_{il} \cdot v_{jl} \]  

wherein \( k \) is a hyperparameter that determines the dimension of the embedding vectors. \( v_{ij} \) is the \( l \)th value in the \( r \)th embedding vector of the user information or the service information.

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

\[ \sum_{i=1}^{f-1} \sum_{j=i+1}^{f} \langle V_i, V_j \rangle x_i x_j = \frac{1}{2} \sum_{l=1}^{k} \left( \left( \sum_{i=1}^{f} v_{il} x_i \right)^2 - \sum_{i=1}^{n} v_{il}^2 x_i^2 \right) \]  

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

\[ \min_{\theta} L(y, \hat{y}) = \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{n} I_{ij} (y_{ij} - \hat{y}_{ij})^2 \]  

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 \( \theta \):

\[ \frac{\partial L(y, \hat{y})}{\partial \theta} = (\hat{y} - y) \frac{\partial \hat{y}}{\partial \theta} \]  

\[ \frac{\partial \hat{y}}{\partial \theta} = \begin{cases} 
1, & \text{if } \theta = w_0 \\
x_{ik}, & \text{if } \theta = w_i \\
x_{ij} \sum_{j=1}^{f} v_{jl} x_j - v_{jl} x_j^2, & \text{if } \theta = v_{jl}
\end{cases} \]  

The following formula is used to update the model parameters:

\[ \theta = \theta - \eta \cdot (\hat{y} - y) \frac{\partial \hat{y}}{\partial \theta} \]  

where \( \theta \) is the model parameters (e.g., \( w_j, v_{jl} \)), \( \eta > 0 \) is the learning rate, and \( \eta \) 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 \( I \) in the formula \( \sum_{j=1}^{f} v_{jl} x_j \) in the first time when iterating parameters, because \( x_{ij} \sum_{j=1}^{f} v_{jl} x_j - v_{jl} x_j^2 \) associates with \( I \) closely. And then, all gradients of \( v_{jl} \) can be obtained easily. Evidently, the complexity of calculating entire \( I \) 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 \( k + f + 1 \) 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/wsdream1.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.

<table>
<thead>
<tr>
<th>Statistics</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Service Users</td>
<td>339</td>
</tr>
<tr>
<td>Number of Web Services</td>
<td>5825</td>
</tr>
<tr>
<td>Number of Web Services Invocations</td>
<td>1,974,675</td>
</tr>
<tr>
<td>Range of Response Time</td>
<td>0-20s</td>
</tr>
<tr>
<td>Range of Throughput</td>
<td>0-1000kbps</td>
</tr>
<tr>
<td>Number of Users ASs</td>
<td>137</td>
</tr>
<tr>
<td>Number of Users Countries</td>
<td>31</td>
</tr>
<tr>
<td>Number of Web Services ASs</td>
<td>1021</td>
</tr>
<tr>
<td>Number of Web Services Countries</td>
<td>74</td>
</tr>
</tbody>
</table>

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).

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

\[
MAE = \frac{\sum_{ij} |\hat{r}_{ij} - r_{ij}|}{N},
\]

and the mathematical expression of NMAE is

\[
NMAE = \frac{MAE}{\sum_{ij} r_{ij}/N}.
\]

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
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 vector is another parameter affecting the performance of the LANFM model. It decides how many factors are used to

<table>
<thead>
<tr>
<th>QoS Properties</th>
<th>Methods</th>
<th>Matrix Density=10%</th>
<th>Matrix Density=20%</th>
<th>Matrix Density=80%</th>
<th>Matrix Density=90%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>MAE</td>
<td>NMAE</td>
<td>MAE</td>
<td>NMAE</td>
</tr>
<tr>
<td>Response Time (0-20s)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UIPCC</td>
<td>0.5842</td>
<td>0.6433</td>
<td>0.4514</td>
<td>0.4970</td>
</tr>
<tr>
<td></td>
<td>LACF</td>
<td>0.5612</td>
<td>0.6181</td>
<td>0.4778</td>
<td>0.5262</td>
</tr>
<tr>
<td></td>
<td>RegionKNN</td>
<td>0.5491</td>
<td>0.6048</td>
<td>0.5155</td>
<td>0.5677</td>
</tr>
<tr>
<td></td>
<td>PMF</td>
<td>0.4865</td>
<td>0.5361</td>
<td>0.4305</td>
<td>0.4743</td>
</tr>
<tr>
<td></td>
<td>NMF</td>
<td>0.4774</td>
<td>0.5261</td>
<td>0.4269</td>
<td>0.4703</td>
</tr>
<tr>
<td></td>
<td>NIMF</td>
<td>0.4792</td>
<td>0.5281</td>
<td>0.4202</td>
<td>0.4630</td>
</tr>
<tr>
<td></td>
<td>LBR</td>
<td>0.4806</td>
<td>0.5293</td>
<td>0.4301</td>
<td>0.4737</td>
</tr>
<tr>
<td></td>
<td>HMF</td>
<td>0.4815</td>
<td>0.5302</td>
<td>0.4298</td>
<td>0.4732</td>
</tr>
<tr>
<td></td>
<td>EFMPred</td>
<td>0.3878</td>
<td>0.4083</td>
<td>0.3332</td>
<td>0.3606</td>
</tr>
<tr>
<td></td>
<td>LANFM</td>
<td>0.3607</td>
<td>0.3966</td>
<td>0.3247</td>
<td>0.3571</td>
</tr>
<tr>
<td></td>
<td>Improve</td>
<td>6.98%</td>
<td>2.85%</td>
<td>2.55%</td>
<td>0.97%</td>
</tr>
<tr>
<td>Throughput (0-1000kbps)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UIPCC</td>
<td>22.3274</td>
<td>0.4700</td>
<td>18.8646</td>
<td>0.3966</td>
</tr>
<tr>
<td></td>
<td>LACF</td>
<td>19.4303</td>
<td>0.4087</td>
<td>16.4495</td>
<td>0.3459</td>
</tr>
<tr>
<td></td>
<td>RegionKNN</td>
<td>24.8487</td>
<td>0.5226</td>
<td>24.0169</td>
<td>0.5050</td>
</tr>
<tr>
<td></td>
<td>PMF</td>
<td>15.9794</td>
<td>0.3362</td>
<td>13.9052</td>
<td>0.2924</td>
</tr>
<tr>
<td></td>
<td>NMF</td>
<td>15.5678</td>
<td>0.3275</td>
<td>13.5386</td>
<td>0.2847</td>
</tr>
<tr>
<td></td>
<td>NIMF</td>
<td>15.1393</td>
<td>0.3185</td>
<td>13.1799</td>
<td>0.2772</td>
</tr>
<tr>
<td></td>
<td>LBR</td>
<td>15.4431</td>
<td>0.3248</td>
<td>13.6455</td>
<td>0.2869</td>
</tr>
<tr>
<td></td>
<td>HMF</td>
<td>15.7076</td>
<td>0.3304</td>
<td>13.5961</td>
<td>0.2859</td>
</tr>
<tr>
<td></td>
<td>EFMPred</td>
<td>13.2966</td>
<td>0.2811</td>
<td>11.4295</td>
<td>0.2417</td>
</tr>
<tr>
<td></td>
<td>LANFM</td>
<td>12.8365</td>
<td>0.2714</td>
<td>11.0560</td>
<td>0.2338</td>
</tr>
<tr>
<td></td>
<td>Improve</td>
<td>3.46%</td>
<td>3.45%</td>
<td>3.27%</td>
<td>1.79%</td>
</tr>
</tbody>
</table>
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.
complexity.

The dimension is 1400. Although we can also obtain the values from 1400 to 1800. The first minimum can be obtained when the dimension is around the local minimum, when varying the dimension gradually converges. The MAE and NMAE values oscillate through the MAE and NMAE values decline quickly, and then it eventually converges. 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_data_set1.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


