Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 1213221, 16 pages
https://doi.org/10.1155/2022/1213221

Research Article

JSTC: Travel Time Prediction with a Joint Spatial-Temporal Correlation Mechanism

Alfateh M. Tag Elsir, Alkilane Khaled, Pengfei Wang, and Yanming Shen

School of Computer Science and Technology, Dalian University of Technology, Dalian, China

Correspondence should be addressed to Pengfei Wang; wangpf@dlut.edu.cn

Received 13 February 2022; Revised 27 March 2022; Accepted 20 April 2022; Published 23 May 2022

Academic Editor: Lijun Sun

Accurate travel time prediction is one of the most promising intelligent transportation system (ITS) services, which can greatly support route planning, ride-sharing, navigation applications, and effective traffic management. Several factors, like spatial, temporal, and external, have big effects on traffic patterns, and therefore, it is important to develop a mechanism that can jointly capture correlations of these components. However, spatial sparsity issues make travel time prediction very challenging, especially when dealing with the origin-destination (OD) method, since the trajectory data may not be available. In this paper, we introduce a unified deep learning-based framework named joint spatial-temporal correlation (JSTC) mechanism to improve the accuracy of OD travel time prediction. First, we design a spatiotemporal correlation block that combines two modules: self-convolutional attention integrated with a temporal convolutional network (TCN) to capture the spatial correlations along with the temporal dependencies. Then, we enhance our model performance through adopting a multi-head attention module to learn the attentional weights of the spatial, temporal, and external features based on their contributions to the output and speed up the training process. Extensive experiments on three large-scale real-world traffic datasets (NYC, Chengdu, and Xi'an) show the efficiency of our model and its superiority compared to other methods.

1. Introduction

Travel time forecasting (TTF) has been considered as one of the most essential services in intelligent transportation systems (ITSs), which greatly supports route planning, ride-sharing, navigation applications, and effective traffic management. TTF is widely used throughout location-based applications and has become one of the most important services in these applications. However, producing an accurate TTF is still challenging since understanding the effects of different dynamic factors (such as urban flows, jams, peak hours, and special situations like public holidays, events, and vacations) on the travel time is a complex task [1]. The dynamic factors can be categorized into four groups as follows:

(1) Spatial dependencies: travel time is greatly affected by the traffic conditions of each region and its neighbors as well, so trips from areas with heavy traffic will take a longer time than others.

(2) Temporal dependencies: traffic conditions during different periods of the day affect the time of travel. For example, road traffic congestion in downtown cities is more severe during the morning and evening peak hours.

(3) Periodical dependencies: periodic patterns such as working hours, weekends, and public events can also affect travel time, where traffic is more congested during workdays and peak times, for example.

(4) External factors: several external factors have also a big impact on the travel time fluctuations, such as weather, holidays, and public events.

Due to the complexity of the spatiotemporal correlations, TTF is a very challenging problem, so accurately

predicting travel time has become a vital task recently [2, 3]. In general, the TTF has been treated as one of two methods (route-based and OD-based) using statistical methods, classical machine learning, and deep learning approaches. First, for route-based approaches, GPS and time series datasets of trajectories are useful in estimating travel times for both road segments and the entire path. However, some complex issues in this technique lead to inaccurate results and costly computations, such as sparsity in trajectory data and GPS devices' errors. Second, the OD-based approach is completely based on the shortest path between the origin and destination points, which reduces the heavy computations and minimizes accumulated error rates of GPS devices. Therefore, the aim of this work is to provide a solution that improves the forecasting accuracy of the OD-based travel time. Many methods have been proposed for TTF, including linear regression (LR) [4], time-varying [5], Kalman filtering (KF) [6, 7], autoregressive integrated moving average (ARIMA) [8], seasonal ARIMA (SARIMA-KF) [9, 10], and random forest (RF) with gradient boosting (GB) (RF-GB) [11]. However, the major disadvantage of these approaches is that they are inappropriate for capturing the relationships between the complicated traffic factors. Most recent researchers have proposed deep learning models that strive to enhance TTF results, such as backpropagation neural networks (BP-NNs) [12–14], long short-term memory (LSTM) [15, 16], convolutional neural networks (CNNs) combined with LSTM (CNN-LSTM) [17], and attention mechanism [18].

Unfortunately, these approaches still suffer from some difficulties, e.g., time-consuming and low speed during the training process, so these methods cannot perform concurrent processing. The sparsity of traffic data represents another concern of TTF approaches, where the historical traffic data do not cover the entire region. On the other hand, the correlations between the spatial features have been considered in many existing works, but most of these methods only focused on the local spatial correlations with the observance of the GPS coordinate points' nearby relationships [19–21]. Sometimes there may not exist similar records with the same location in the historical traffic data. Therefore, we attempt to solve this issue by considering the records of distant neighbors. Besides, nearby regions can be relevant and very similar in terms of traffic patterns during various periods. Herein, finding a mechanism capable of integrating relevant spatial and temporal features and simultaneously capturing the complicated dependencies between them can be very helpful. The supplementary critical factors play a significant role in traffic pattern fluctuations, especially within the extreme circumstances of these factors as examples (weather conditions, public holidays, events, and vacations). Thus, we model these features according to the features' correlations and dependencies between each other and also consider the features' contributions to the output. The main contributions of our work can be summarized as follows:

(i) Since data sparsity is a key challenge in real traffic scenarios, we propose a method to solve this issue and achieve better results by splitting the city into N × N grids using geo-hashing techniques and dividing the city into different clusters using the K-means algorithm. This allows us to use neighboring trips if there are no historical records or if the historical records are insufficient.

(ii) We propose a new mechanism to capture both spatial and temporal dependencies. This mechanism comprises two modules: the spatial self-attention module (SSAM) that is used to infer the spatial relationships and the residual dilated convolutional module (RDCM) to capture dynamic time dependencies.

(iii) Moreover, we adopt a multi-head attention approach to learn the attentional weights of a multimodality factor (spatial, temporal, and external) based on their contributions to the target. While many previous works use RNNs in their models, which are time-consuming in the training stage due to their recurrent nature, we use a multi-head attention mechanism that supports parallel computing in this work to dramatically reduce training time.

(iv) We conduct extensive experiments using three large-scale traffic datasets in three different cities (NYC, Chengdu, and Xi'an). The results demonstrate the efficiency of our model compared to other methods under various traffic conditions.

The rest of this paper is organized as follows. Section 2 reviews the related works about the TTF approaches. Section 3 contains the problem definition and formalization, followed by data processing and analysis. Thereafter, we describe our proposed framework (JSTC) in detail. Section 4 discusses the experimental results of our model compared to other models. Finally, a summarized conclusion of this paper is presented in Section 5.

2. Related Work

Generally, TTF methods can be classified into two categories: route-based and OD-based methods.

2.1. Route-Based Methods. Route-based methods can be divided into two approaches.

2.1.1. Segment-Based Method. This method divides the road into segments and then estimates the travel time for each segment individually. Finally, the total travel time for the entire path is the summation travel time of all segments [22, 23]. Many researchers consider the TTF as time series forecasting for a single road, such as the ARIMA model and KF [24, 25], which have been applied in short-term forecasting for road section travel time. In addition, support vector regression (SVR) was used due to its competence and generalization compared to the historical average (HA) method [26]. The gradient boosting decision tree method (GBDT) has been also used to improve prediction accuracy

on TTF problems [27]. Wang et al. [15] investigated the sequence relationship between the road segments. They treated the travel time of the segment as a sequence of time series data and then used the LSTM model to solve this sequence prediction problem. The spatiotemporal hidden Markov method (STHM) was also applied to capture the correlations among different traffic time series and then predict the travel time [28].

2.1.2. Path-Based Method. Another group of researchers combined multiple route segments as an entire path instead of using one road segment to solve the TTF problem. This considers the impact of intersections and traffic lights, which leads to more accurate predictions in the path-based method [21, 29]. A non-parametric technique for route TTF based on floating car data (FCD) is the first to use the path-based approach [30]. It accumulated the travel time of each road segment from a low frequency instead of calculating the travel time of the subpath. Rahmani in [31] also proposed a route-based method for route TTF by combining multi-traffic data sources collected by FCD and automated number plate recognition (ANPR). In [32], the K-shortest path algorithm was developed to infer the possible paths from each OD trip and then predict the link travel time. However, these techniques frequently suffer from dispersed data or the high cost [21]. Nowadays, vast amount of taxi trajectory data is collected by GPS equipment, so the TTF model for a direct path was proposed based on a three-dimensional tensor by applying two essential components; first, compute the travel time for each segment by the tensor decomposition. Then, find the most optimal elements that help to estimate the route's travel time [33, 34]. In [35], a deepIST model was proposed that takes spatial and temporal dependencies of traffic patterns into account by using map image information of the trajectory to predict travel time. In this framework, two CNN-based modules were combined to make images of the route segments and then look for spatial and temporal traffic correlations. To address the data sparsity issue that may occur in some trajectory segments, a CNN with LSTM model named DeepTTE was proposed for raw trajectory data processing [17].

2.2. OD-Based Method. Many scholars have chosen the OD-based methods to address the TTF issues, to minimize the time needed and avoid the complex computations and complicated implementation. In [20], the authors proposed a multi-task representation learning model (MURAT) based on OD data, which achieved promising results. However, this method requires a long processing time and needs a lot of data, which seems to be the main disadvantage of this model.

The estimation of the average time of the urban routes based on the candidates' paths expected between OD trip coordinates was proposed in [19, 32]. They combined the trucks' and taxis' travel datasets to predict travel time between each grid zone, followed by the same methodology in [32], while Faruk in [36] were the first scholars to develop a model for the TTF based on travel distance predicted directly through the OD coordinates' GPS data. However, they ignored delays in intersection queuing, which can reduce the TTF prediction precision. Recently, an ensemble technique with a multi-modality data source model named TTE-Ensemble was proposed in [21]. In this model, the ensemble method was adopted with GBDT and DNN models. GBDT and DNN predicted the travel time separately. Then, each models' results are fed to a decision tree algorithm as a meta-learner model to achieve the final TTF for each OD trip. However, this model basically relied on converting the trajectory data into 2D square cells instead of real OD locations which means that all trips with the same grid ID will have similar characteristics regardless of their distance. Nevertheless, the GBDT and decision tree approaches are unsuitable for big data due to the high computational cost.

Recently, the attention mechanism has been widely used for traffic forecasting. In [37], the authors proposed a pairwise self-attention mechanism for capturing the spatial and temporal dependency of traffic flow prediction. In [18], a deep learning model named FMA-ETA was proposed, which predicted travel time by combining a feed-forward network and self-attention. This model focus on spatial dependencies while temporal correlations were ignored. Besides, convolutional and graph neural networks have been used for spatial and temporal correlations in traffic speed forecasting [38]. A model called GSTGCN, which applies dilated convolutional network architectures to take the advantage of dilation rate by increasing covered spaces between the inputs, was designed.

The literature survey concluded that most of the previously discussed methods did not completely handle the TTF issues and achieve high accuracy due to the complexity of spatial-temporal correlations learning, considering the differentiation of the road network topology and extreme temporal conditions. Also, there are some techniques that could be beneficial for improving the accuracy of travel time prediction. Inspired by the aforementioned ideas, we propose a JSTC framework relying on OD-based strategy, which can achieve high accuracy with promising performance in predicting the travel time for any given OD GPS points. Herein, our work mainly addresses the sparse spatial data problem and also focuses on the multi-component correlations between spatial, temporal, and external factors, which significantly affect the travel time.

3. Methodology

The aim of the traffic forecasting task in this paper is to predict travel time between any pair of locations by means of the observed historical traffic datasets. The general overview of our methodology mainly consists of three main parts: data preparation and preprocessing, analysis of traffic pattern similarity, and introducing our proposed model in detail. To begin, data preparation and preprocessing are critical, which include data cleaning and removal of noise and outliers, feature extraction, and geo-localization (clustering and grid-partitioning). Then, we get through the spatial and temporal dependencies' similarity investigation to observe the influence of these components in traffic patterns' fluctuation.

Finally, we introduce our prediction model, which aims to predict the total travel time of the OD trips accurately. The detailed descriptions of each of these parts are given in the following sections. In advance, we formalize the traffic forecasting problem in this work as in the following key concepts and definitions.

3.1. Preliminaries. We define and formalize the TTF problem as a travel time prediction task between two given points (A) and (B).

Definition 1. OD-trip P $_i$: We define a trip from the historical records as (P), which consists of 5-tuples (o, d, t, **D**, T), where (o) is the pickup location (A), while (d) is the drop-off location (B). Also, (t) denotes the trip time-stamp, which includes the pickup and drop-off times as (t_o) and (t_d), respectively. Both the origin (A) and destination (B) are 2-tuple GPS coordinates, as $o_i = (olat_i, olong_i)$ and $d_i = (dl\ at_i, dl\ ong_i)$, where trip distance (D $_i$) can be obtained from these coordinates. To find the matched historical trips for trip P_i, we define a query (Q) as follows:

$$Q = \{P_i\}_{i=1}^{N}. \tag{1}$$

Definition 2. Spatial and temporal tensors: after splitting a city into N × N grids (G) and K-clusters (C) as a geo-region based on the OD-GPS coordinates, the GPS points have been mapped into G and K as well. We define two 3D tensors $\delta^i \in P^{H_\delta \times F_\delta \times 1}$ and $\tau^i \in P^{H_\tau \times F_\tau \times 1}$ to represent spatial features (δ^i) including pick-up locations, drop-off locations, speed, distance, cluster-ids, grid-ids, and other auxiliary features. Besides, temporal (τ^i) features include the day of the week in-between (0–6), the hour of the day in-range (0–23), and the day of the month as (0–30), where H represents the historical record ID and F denotes spatial or temporal features. Note that we consider the trip features as sequence.

Definition 3. TTF for trip P $_i$: we define the travel time T_i as the total time for the trip P $_i$ from (A) to (B) as follows:

$$T_i = \left[t_{d_i} - t_{o_i}\right]. \tag{2}$$

Hence, the main goal of our work is to estimate the total time (T_i) for an OD-trip (P $_i$) with an assist from the historical trips by a query (Q).

3.2. Data Analysis and Preprocessing. In this paper, we used three large-scale real-world traffic datasets (NYC, Chengdu, and Xi'an) to verify the efficiency of our model across various road network topologies and traffic patterns. The first dataset is the NYC taxi dataset, which is provided by the New York City Taxi and Limousine Commission (TLC) [39] with billions of trip records from 2009 until now and comprises 21 different variables, including GPS coordinates for pick-up and drop-off, pick-up and drop-off time-stamp, total trip distance in miles, and other features. Following [40], we extracted six months of the traffic data between 01/01/2016 and 30/06/2016 for analysis and experiments in our work. The data we have selected contain approximately 75 million records, with over 12 million trips per month and 416,666 trips per day. The other two datasets are Chengdu and Xi'an, which were provided by the "Didi Chuxing platform" containing 9,707,970 and 5,272,758 taxi trajectories in September and October 2018 for Chengdu and Xi'an, respectively. The average trip per day is (123,463 and 133,843) trips, respectively (Table 1).

The analysis of traffic data can greatly assist in recognizing the fluctuations in traffic patterns. Spatiotemporal data cleaning and anonymous value filtration were conducted by removing the invalid or uncharted trips' records that contain missing information in one or more parts of OD GPS location, passenger count, and pick-up/drop-off interval-time records. We consider the trips out of the city boundary as spatial outliers and clean them accordingly. Also, all trips with a distance less than 500 meters and more than 100 kilometers have been cleaned. The temporal components have been filtered by taking only the records with the travel time less than 24 hours (86,400 seconds) and over 3 minutes (180 seconds). In order to observe the traffic patterns over the whole city, 15 regions were classified according to the city's boundaries. Then, each region was grouped by temporal dependencies (day of month and day of week) to obtain the similarity of week and day rhythms. Considering the time-interval of the day as (0–23), we measured the average rate of travel time for all trips within the same spatial and temporal information, as well as traffic intensity for all trips that flow in and flow out across these regions. Figure 1(a) represents the average rate of trip density, and we can see a low-density rate in the period from midnight up to 6 AM. In contrast, we can notice that the maximum density rate happens during two peak periods, from 7 AM to 9 AM as morning rush hours and from 6 PM to 8 PM as the evening rush period. For example, during the early morning and evening rush hours, there is heavy traffic congestion that means the movement will be slow. Therefore, through the non-peak hours, traffic patterns seem to be normal. Note that the average rate of travel time in Figure 1(b) is quite similar to the density rhythm in terms of increase and decrease rate, except for trips with a long duration. So, each trip was considered as one counted trip in the density rate computation, whereas the trip's duration was taken into account while calculating the average rate of travel time, which affects the total average time in this case. Moreover, to determine peak and non-peak periods for Chengdu and Xi'an cities, we did some statistical analysis over various given regions within the same conditions. We randomly chose regions to illustrate the influence of traffic patterns. Table 1 shows that the average traffic volume measured (historical records which enter or leave the cluster or grid) is probably relatively low or high, especially in areas with heavy activity. The results show that the average travel time varies from one region to another according to the traffic rhythms during the hours of the day. On the other hand, traffic density during morning and evening hours is much higher than night and afternoon hours, which explains that traffic overcrowding influences traffic speed and travel time.

Table 1: Statistical analysis of the traffic patterns and fluctuations in particular locations in Xi'an and Chengdu cities.

Day	10 Oct #Xi'an⟍#Chengdu	13 Oct #Xi'an⟍#Chengdu	15 Oct #Xi'an⟍#Chengdu
Pick_Hour	9, 11, 20	8, 10, 16	7, 14, 19
Pick_location_Grid	136⟍20	136⟍20	136⟍20
Drop_location_Grid	39⟍38	39⟍38	39⟍38
Avg_traffic_volume/grid	2593, 2301, 2311⟍1610, 1381, 1370	1470, 1633, 2177⟍1551, 1338, 996	2368, 1405, 2563⟍1457, 1238, 706
Trip_distance (km)	~ 8.5⟍ ~ 5.3	~ 8.5⟍ ~ 5.3	~ 8.5⟍ ~ 5.3
Trip_speed (kms)	30.7, 33.3, 57.06⟍19.1, 12.1, 17.5	39.1, 33.8, 23.9⟍10.3, 19, 26.5	24.5, 41.5, 52.6⟍11.3, 19.1, 23.9
Trip_duration (sec)	1372, 1368, 744⟍1178, 1986, 1324	1076, 1242, 1761⟍2187, 1178, 951	1719, 1014, 798⟍2394, 1263, 923

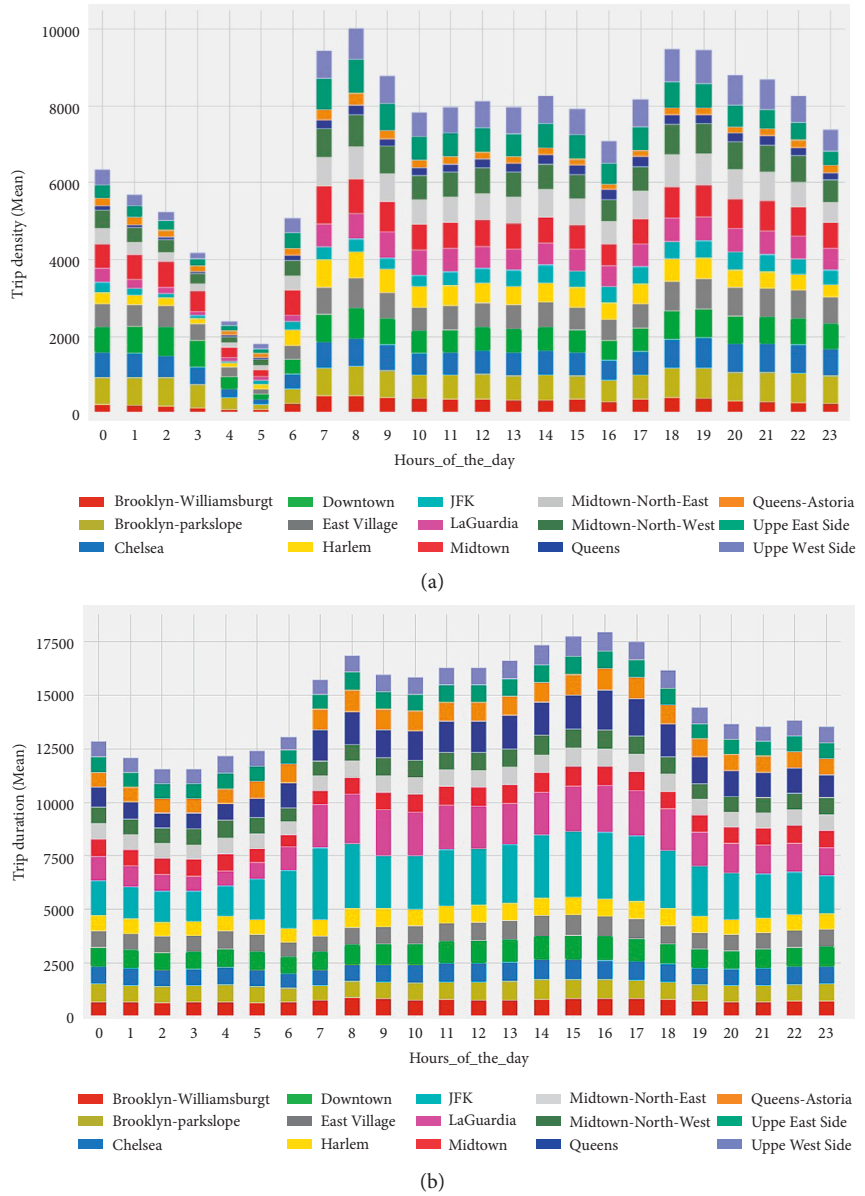

Figure 1: Traffic pattern visualization for city boroughs. (a) Average rate of trip density. (b) Average rate of trip duration.

Eventually, to ensure that our proposed model is capable of producing effective results, after investigating the traffic patterns' similarities, two peak periods have been adopted for NYC, Chengdu, and Xi'an as the morning and evening peak periods, which include (7 ~ 10 AM and 5 ~ 8 PM), respectively.

3.3. Feature Extraction and Data Preparation. Similar to [21], we apply data preprocessing based on the perspective of multi-modality. Thus, accurate prediction of TTF is greatly influenced by numerous dynamic components, including complicated spatial and temporal dependencies, and the influence of external factors such as weather status, social events, or public holidays [41, 42]. Hence, to improve the prediction accuracy, we adopted three components in our proposed method: spatial, temporal, and external. We adopt two 3D tensors δ^i and τ^i for spatial and temporal components' representation, while the external components were divided into two subvectors: weather data and public holiday data.

3.4. Spatial Components. The original dataset provides the trips' pick-up and drop-off GPS locations only, so we further extracted additional spatial features from these two points such as distance and speed, which are essential spatial features. We applied two different methods to calculate the distance between two GPS locations. The two methods are the Manhattan and haversine distance approaches [40]. Manhattan distance is formulated as follows:

$$\left|\triangle \mathrm{lat}_{P_i}\right| + \left|\triangle \mathrm{lon}_{P_i}\right|, \tag{3}$$

where $\triangle \mathrm{lat}_{P_i}$ and $\triangle \mathrm{lon}_{P_i}$ denote the total distance difference between the ordered pairs of OD coordinates computed by the following equations:

$$\begin{aligned} \triangle \mathrm{lat}_{P_i} &= \left|o_i\mathrm{lat} - d_i\mathrm{lat}\right|, \\ \triangle \mathrm{lon}_{P_i} &= \left|o_i\mathrm{lon} - d_i\mathrm{lon}\right|. \end{aligned} \tag{4}$$

The haversine distance is also formulated as follows:

$$2r\arcsin\sqrt{\sin^2(\triangle\phi/2) + \cos(o_i\mathrm{lat})\cos(d_i\mathrm{lat})\sin^2(\triangle\lambda/2)}, \tag{5}$$

where $(\triangle\phi)$ is $(\triangle \mathrm{lat}_{P_i})$ and $(\triangle\lambda)$ is $(\triangle \mathrm{lon}_{P_i})$.

Furthermore, the average speed was calculated regarding the trip distance and trip duration. In addition, we extracted other supplementary spatial features from the GPS coordinates, for example, cluster and grid density, which are explained in Definitions 4 and 5, respectively. In the real-world road network, traffic patterns' variation is highly related to time (e.g., traffic tidal phenomena during the weekdays) and space, including neighboring regions. Thus, the traffic patterns in neighboring regions are more relevant. Generally, traffic in neighboring regions exhibits similar flows over the day-time intervals.

To improve the proposed model's performance, we applied the K-means clustering method in the spatial component preprocessing phases. Since K-means attempts to group places based solely on their Euclidean distance, it returns clusters of places that are close to each other and geo-positioning trips within nearby regions into the same cluster. In order to determine whether we are using the right number of clusters, we applied the elbow curve method [43] based on calculating the sum of squared errors (SSE) for a range of values of k (60, 80, 100, 120, and 150) and then picking the elbow of the curve as the optimal number of clusters to use by choosing a small value of k that still has a low SSE. From Figure 2, we can observe that the optimal value of K is 100.

Similarly, we mapped each OD-trip into 2D**D** grid cells with an area of approximately 0.5 km × 0.5 km. Thus, we can represent each trip with two grid-ID features, one for pick-up and the other for drop-off. Finally, after the clustering and geo-location mapping processing, the degree of crowding for each part (cluster and grid) throughout the city is computed depending on the following definitions.

Definition 4. Density score for cluster:

$$\mathrm{Cluster}_{\mathrm{density}}(d_C) = \sum_{i=1}^{N} o_{C_i} + \sum_{i=1}^{M} d_{C_i}. \tag{6}$$

Definition 5. Density score for grid cell:

$$\mathrm{Grid}_{\mathrm{density}}(d_G) = \sum_{i=1}^{N} o_{G_i} + \sum_{i=1}^{M} d_{G_i}, \tag{7}$$

where N and M represent the total number of origin (o) and destination (d) trips' locations recorded within the same cluster (C) and grid (G) at time interval of the day. These two spatial features are essential to reflect the traffic flow of the region through different periods.

3.4.1. Temporal Components. The temporal features are significant factors to understand travel time changes through time variation. Therefore, trip duration is affected by several temporal factors, which may occur daily, weekly, or seasonally [44]. The rhythm of commuters' flow over workplaces, schools, and even public places is an example of activities that cause traffic jams at various times. To this end, the following temporal features were extracted from the traffic datasets, using the one-hot encoding (OHE) and label-encoding techniques as follows:

(i) We represent the day of the month as a label value from 0 to 30.

(ii) We represent weekdays as a categorical value from 0 to 6.

(iii) We represent hours of the day as a label value from 0 to 23.

(iv) Working days and weekends take 0 or 1.

3.4.2. External Components. The external factors were divided into two parts: weather conditions and public holiday. Generally, the trip is affected by one or more of the following weather conditions (heavy rain, snow, storms, and so on). Different weather conditions can also result in varying travel times with similar spatial patterns and different interval times. Hence, the weather is considered as an important external factor in this work. Table 2 shows the weather data categories, which are classified into 10 types (sunny, cloudy, rainy, windy, and so on). Also, three more features are used

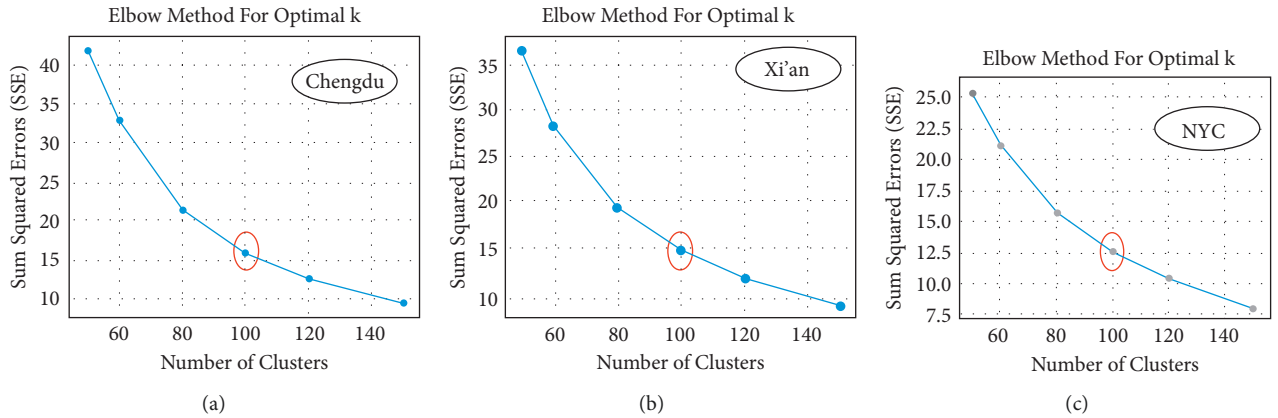

Figure 2: Investigation of selecting K-means clustering value using elbow curve method.

Table 2: Weather data formalization and labeling.

Weather condition	Label
Clear	1
Overcast, partly cloudy, mostly cloudy, scattered clouds	2
Haze, fog	3
Light freezing fog, light freezing rain,	4
Light snow	5
Rain	6
Snow	7
Heavy rain	8
Heavy snow	9
Light rain, sleet	10

to describe the weather situation of trips circumstances in terms of extreme weather conditions (snowing, raining, or foggy). There are 16 different types of weather conditions, according to the historical weather data provided in [45]. Thus, this classification process makes similar weather conditions much closer and helps to reduce the data dimensions. Because of variable weather conditions, the same spatial locations in terms of OD-grids may not have the same trip times, as shown in Figure 3. This figure shows that when the weather is regular, travel time between the same origin and destination grids takes less time than hours characterized by extreme weather conditions when comparing two different days.

Besides the factors mentioned above, the traffic patterns during public holidays and events can differ from those of the daily routine, due to increased outdoor activities or variation in daily traffic patterns, leading to extreme traffic jams. As a result, two subcategorical features are concluded from the NYC and China public holiday datasets to represent whether the day is a holiday or not. Eventually, externals are classified into two types: categorical features by using the OHE technique and discrete features. Furthermore, data standardization and scaling techniques for features have been utilized.

3.5. JSTC Model Architecture. Our proposed framework mainly comprises three modules, as shown in Figure 4. The first block is designed to learn the dependencies between spatial and temporal components and capture their complicated relations. This block also helps to capture the correlation between grids and clusters for OD-trips during different time patterns, especially when observing adjacent locations' properties and dealing with the sparse data. After processing the external features, we combine all feature representations and pass them to the last block, which is the multi-head attention module to learn the attentional weights of all features based on their contribution to the output. Next, we describe each part in detail.

3.5.1. Spatial Self-Attention Module. In this section, we develop a self-convolutional attention mechanism that captures the correlations across different spatial features and learn their attentional weights. To this end, we adopt a 1D convolutional layer followed by self-attention heads. Figure 5 shows our proposed spatial self-attention module, and the spatial feature's tensor includes a pair of GPS coordinates, a pickup cluster, a drop-off cluster, a pickup grid, a drop-off grid, distance, and speed {D and S}. First, we reshape the input into three dimension as an input for the 1D convolutional layer. To do so, we used a reshape function to reshape the 2D features vector into 3D tensor δ^i. Then, we used the convolution filter and kernel size as shown in Figure 5 to handle the spatial input tensor. Thus, we can get *Query* $\{Q^{\delta}\}$, *Key* $\{K^{\delta}\}$, and *Value* $\{V^{\delta}\}$ as an output from each 1D-Conv layer followed by the ReLU activation function as follows:

$$Q^{\delta} = \omega_q^f \cdot \chi^{\delta}, \quad K^{\delta} = \omega_k^f \cdot \chi^{\delta}, \nu^{\delta} = \omega_V^f \cdot \chi^{\delta},$$
$$\text{Conv1}d_{(K,Q,V)^{\delta}} = \chi_{i,j}^{\delta} = \sum_{j=1}^{J} \omega_f^{(j)} \odot \chi_{(i+\kappa)} + \beta^j, \tag{8}$$
$$\text{ReLU}(\chi) = \text{Max}(0, \chi),$$

where χ denotes the tensor input, i is the convolution processed index, j refers to the filter (f) position, and κ is the kernel size. (ω_f^j) represents the filter (f^j) weight matrix, and (β^j) is the learnable parameter (bias). We set

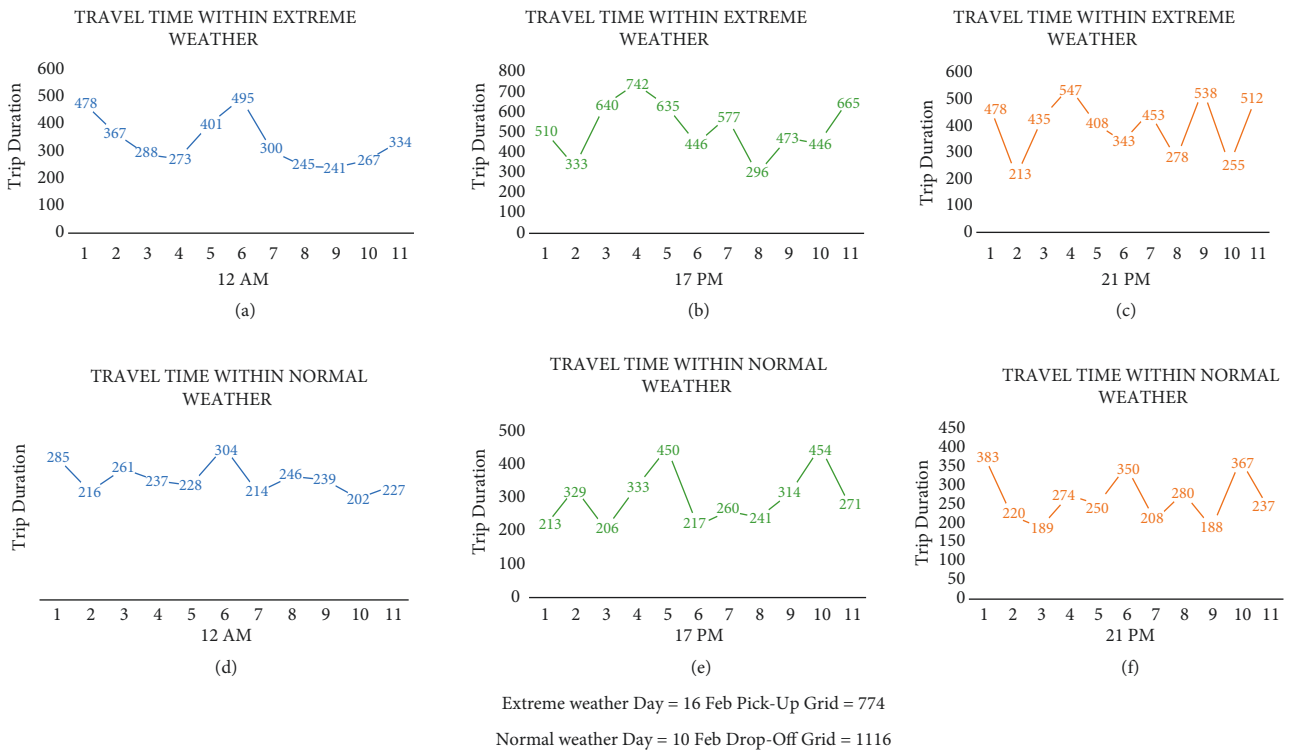

FIGURE 3: Visualization of travel time changes when the weather is different, for example, from pick-up grid-id = 774 to drop-off grid-id = 1116 on the NYC dataset on two different days over the same time interval.

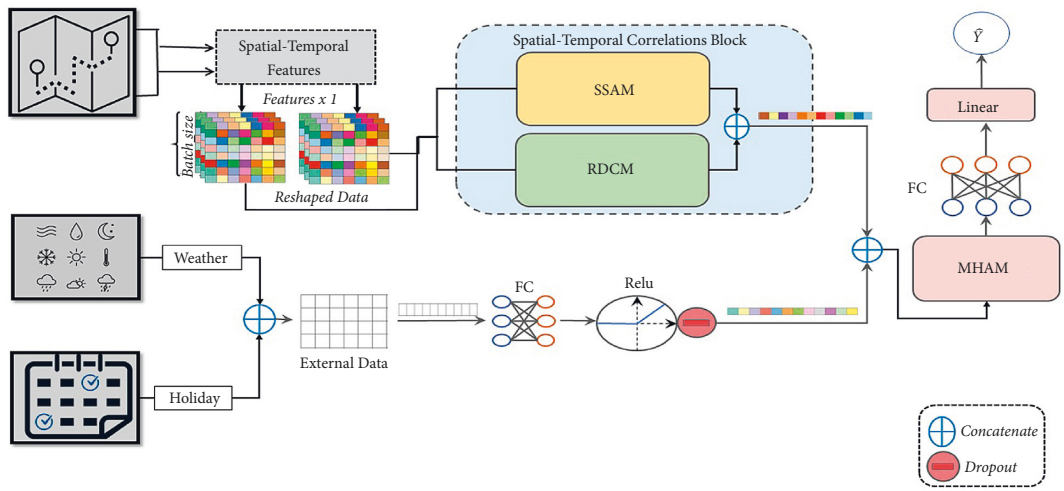

FIGURE 4: Joint spatial and temporal correlation (JSTC) mechanism architecture combines spatiotemporal correlation block, which includes the spatial self-attention module (SSAM) and residual dilated convolutional module (RDCM). Then, we used a multi-head attention module (MHAM).

the filter and kernel size to 1 and 3, respectively. We set the padding to "same" to avoid dropping some information and verify that all inputs are completely represented. Therefore, the weight matrix $(\breve{U})$ between K^{δ} and Q^{δ} is computed by using the scaled dot attention function, and then the final attention score $(\breve{W}^{\delta})$ is computed as in the following equation:

$$\begin{aligned} \breve{U} &= K^{\delta} \odot Q^{\delta}, \\ \breve{W}^{\delta} &= \breve{U} \odot V^{\delta}. \end{aligned} \tag{9}$$

Afterward, the final attention output is obtained over the multiple self (attention) layers, and then we flatten the output of the spatial self-attention block and concatenate it with the temporal correlation output.

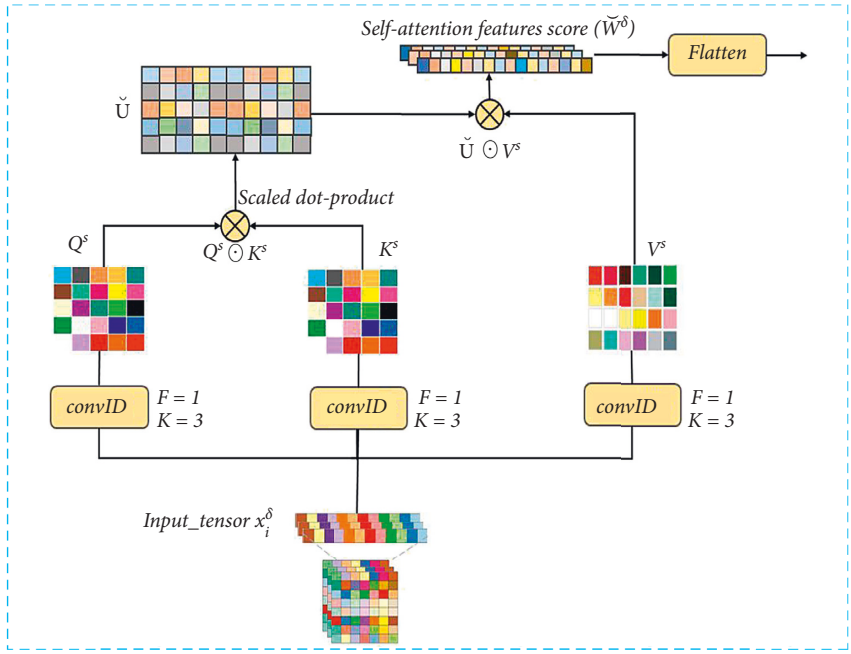

Figure 5: The structure of spatial self-attention module (SSAM).

3.5.2. Residual Dilated Convolutional Module. The temporal convolutional module aims to capture the temporal patterns. Several previous studies have considered the temporal dependencies of traffic forecasting tasks. In [46, 47], the RNN architecture was applied to capture temporal relations, while references [48, 49] utilized the gated recurrent units (GRUs) and long-short memory (LSTM) networks to model the temporal components on traffic pattern fluctuations. Although these approaches have shown good performance, they still suffer from many problems (e.g., exploding/vanishing gradients, time-consuming in the training phase, and some other limitations in modelling long sequences).

Inspired by the recent success of the temporal convolutional network (TCN), we propose a residual temporal correlation module (RDCM), which comprises multiple dilated 1D-Conv layers stacked together as shown in Figure 6. We employed the TCNs advantages in the convolutional operations expanding domain by adjusting the dilation rate parameter on each layer. Empirically, same as the preprocessing we have used for the spatial components, we construct 3D tensor (τ^i) for the temporal features. Since the traffic patterns during the different periods of the day are highly affected by the traffic flow in each region. Accordingly, while investigating the dependencies of temporal factors, some spatial features should be considered due to their significant impact on the output. In our case, the density score grid and cluster for both pick-up and drop-off, which are measured hourly, have been adopted as supplementary features for the temporal correlation modelling. By now, the temporal component of each trip record is represented by the (χ_i^{τ}) tensor, which includes the temporal features and the supplementary features.

In order to capture the interactions and patterns of temporal features in terms of long-short dependencies between the input features, we built three dilated convolutional layers with different "dilation −rates" as = {1, 2, 4} to address the following two key points: avoiding the backpropagation issue (gradient vanishing or exploding) and receptive field expansion to cover the entire input's representation through the shallow hierarchical layers. Thus, to achieve the normal convolution operation, we set the dilation "$d^r = 1$" and the kernel-size "$K = 3$" in the first layer followed by ReLU and drop layers, and then the output is used as an input for the next dilated convolution layer with "$d^r = 2$" and "$K = 3$." Then, "$d^r = 4$" and "$K = 5$" for the last layer. Figure 6(b) illustrates the dilated convolution steps. As a result, we make sure that the different space (long-short) of the relationship between the temporal factors has been considered. Also, an efficient representation of the features without missing any important information is also considered. The dilated convolutional layers were combined into a residual block, and an element-wise concatenation layer was used to add the last output to the input (χ_i^{τ}), which can improve training and maintain an optimal feature correlation distribution. In this paper, we formulated the DRCM block operations as follows:

$$f\left(\tau_i^{dr}\right) = f\left(\chi_i^{\tau}\right) + \chi_i^{\tau},$$
$$f\left(\chi_i^{\tau}\right) = \sum_{s=1}^{S} \left[\chi_i^{\tau} + d^r \circledast s\right] \omega[s], \tag{10}$$

where d^r denotes the "dilation −rate" and s denotes the "filter− size." Eventually, the temporal correlation output is concatenated with the previous spatial correlation outputs and passed to a multi-head attention mechanism.

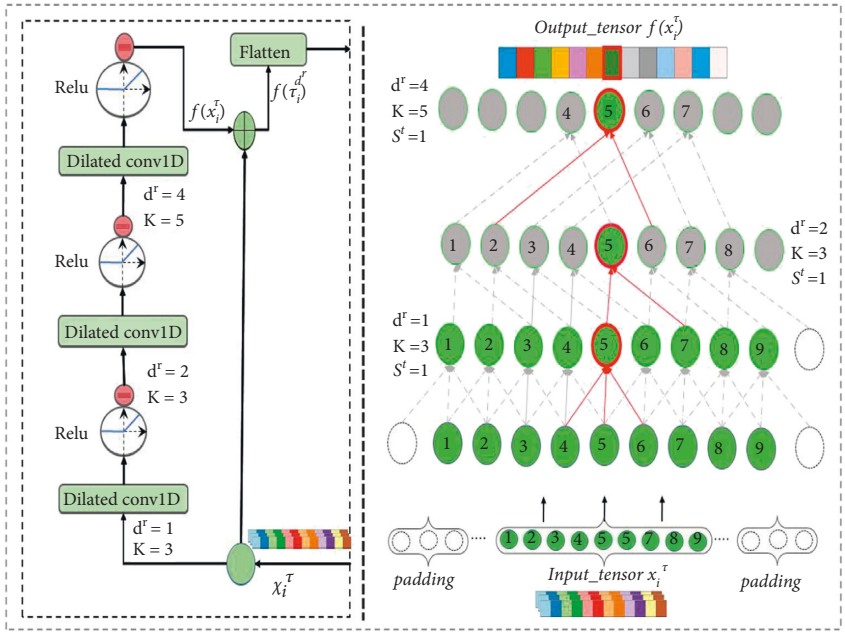

FIGURE 6: An illustration of our proposed residual dilated convolutional module (RDCM). On the left side of the figure, we show the RDCM module's architecture. The right side represents the dilation convolutional operations by expansion of the receptive field (dilation parameter) and different sizes of kernels (K) to obtain optimal feature representations.

3.5.3. Multi-Head Attention Module. The multi-head attention mechanism is illustrated on the right side of Figure 4 as reported in [42], which has been adopted in our model in charge of getting accurate prediction results. First, due to the impact of the external features on the travel time as mentioned before, we apply a fully connected layer followed by ReLU and dropout layers as subblock to represent the external factors (weather details and public holidays), and then we combine the external features' representation vector with the vector that represents the spatial and temporal correlations outputs (for more details, see Sections 3.4.1 and 3.4.2). By implementing this mechanism, we can enhance our model's ability to learn the attentional weights of various features using multiple attention layers. Besides, it makes the training process robust and fast where it guarantees processing strategies across multiple (H_{Att_h}) heads. Thus, from the concept of learning the attentional weights of all features based on their contribution to the output. In this study, the attention scores represent the intercorrelations of the input features to the target (travel time). Therefore, we applied a "scaled-dot" function to compute the attention score based on the contribution of each feature to the output target. To do so, we constructed (query (Q), key (K), and value (V)) vectors, which include the feature representations. Firstly, we can get the features' scores (weights) between each feature in (Q) and the set of keys, and then the second round of dot-product function takes these scores' (weights) vector and set of keys (K) to get the values' (V) vector, for calculating the final attention score. We formally defined this process as follows:

$$\begin{aligned} MH_{\text{Att}}(Q,K,V) &= \text{Concat}\left(H_{\text{Att}_1},\ldots,H_{Att_h}\right)W^{\mathbb{O}}, \\ H_{\text{Att}_i} &= \text{Attention}\left(QW_i^{\mathbb{Q}},KW_i^{\mathbb{K}},VW_i^{\mathbb{V}}\right), \end{aligned} \tag{11}$$

where $QW_i^{\mathbb{Q}}$, $KW_i^{\mathbb{K}}$, and $VW_i^{\mathbb{V}}$ represent the (K, Q, and V) weights for each head and $W^{\mathbb{O}}$ is a combination of scores'/weights' matrix. h is the number of head parameter; after several trials with the h values $\{4, 6, 8, 10\}$, we adopted 6 as the number of attention heads, which leads to fast performance and achieves optimal results.

Eventually, we use a dense layer followed by a linear operation to get the final prediction results $(\hat{y}^i_{OD_\tau})$ ideally as follows:

$$\hat{y}^i_{OD_\tau} = \varphi\left(W_f\chi^i + b_f\right), \tag{12}$$

where (φ) is the linear activation function and (W_f) and (b_f) are learnable parameters.

4. Experimental Results and Analysis

We used three large-scale traffic datasets (NYC, Chengdu, and Xi'an) in our experiment. Section 3.2 describes in detail the data analysis and preprocessing. We randomly split the datasets into 80% for training and 20% for testing. The training set was then divided into two subsets: 70% for model training and 30% for validation. The learning rate values range (0.01, 0.001, and 0.0001), batch size as (128, 256, and 512), dropout values range (0.1, 0.2, and 0.3), and multi-head (h) as (4, 6, 8, and 10). The optimal values for parameters are as follows: the learning rate is 0.001, the number of training epochs and attention heads is (60 and 6), respectively, and batch size is 512. Besides, to reduce overfitting, we applied both the kernel regularizer (L2 norm) and dropout (0.2). Also, we adopted the Adam optimizer as an optimizing function with a linear activation function.

4.1. Evaluation Metrics. To evaluate our model, we use two common prediction metrics.

Mean absolute percentage error (MAPE) is calculated as

$$\text{MAPE} = \frac{100\%}{N}\sum_{i=1}^{N}\left|y^i - \hat{y}^i/y^i\right|. \tag{13}$$

Mean absolute error (MAE) is calculated as

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|y^i - \hat{y}^i\right|, \tag{14}$$

where y^i and $\hat{y}^i$ are the actual and predicted OD-trip durations in seconds, respectively. N indicates the total number of records in the test dataset.

4.2. Results

4.2.1. Comparison of Various Models' Results with JSTC Model. To show the performance efficiency of our model, we compared it with the following models:

(i) LRM: we applied the LR model in [20] with almost all features except the grid and cluster, which have a high dimension and cause overflow.

(ii) XGBoost: a machine learning model widely used for both classification and regression problems. However, XGBoost with a deep tree may lead to better predictions. Following [50], we set the max-depth parameter between 4 and 6 to avoid overfitting.

(iii) LightGBM: the LightGBM model is based on decision tree algorithm with leaf-wise and level-wise. This model is more appropriate for large datasets with large dimension of features [51]. Accordingly, we set the LightGBM parameters same as in [21].

(iv) ST-NN: spatiotemporal-based model was proposed in [19], which combined two DNN modules to predict the trip distance and then used this distance to predict the travel time.

(v) TTE-Ensemble: the collaborative model proposed in [21] combines machine learning and neural network (GBDT and DNN) modules for modelling multi-modality data to predict the OD-trip travel time.

(vi) FMA-ETA [18]: a deep learning model based on a multi-self-attention technique integrated with a feed-forward structure (FFN) for capturing spatial and temporal dependencies and obtaining TTF.

(vii) STTNs [37]: two spatial-temporal blocks are integrated into an approach based on graph neural network and transformer (STTNs), which jointly investigates the dynamic spatial and temporal dependencies to enhance the traffic flow prediction result's accuracy.

Table 3 illustrates our model results compared with other models in terms of MAPE and MAE for the NYC, Chengdu, and Xi'an datasets. The results show that our model outperforms other approaches. As previously mentioned, we divided the comparative models into two parts (ML and DL models). The results of ML (LR, XGBoost, and LightGBM) models show worse accuracy compared with the DL models because these simple statistical ML algorithms have difficulty in modelling the non-linearity relations of complex traffic patterns. We notice that the LR model gives the worst results compared to others (26.12, 24.37, and 25.85) in MAPE and (168.34, 176.33, and 197.14 sec) in MAE for NYC, Chengdu, and Xi'an, respectively. The error rate (MAE) was reduced by (14.4, 14.14, and 9.11 sec) and (18.62, 20.94, and 20.88) with the XGBoost and LightGBM models, respectively. In contrast, our model shows better performance where it reduces the errors by approximately (71.22, 104.4, and 108.26 sec) compared with LR and (56.82, 90.26, and 94.15) on XGBoost, while (52.6, 83.46, and 82.38) for LightGBM on NYC, Chengdu, and Xi'an, respectively.

On the other hand, the ST-NN achieved the lowest results of all the DL models because it only utilizes two MLP blocks. In comparison, our model reduced the errors (MAPE) by at least (~ 7%) on NYC and Chengdu, while 6.31% on Xi'an. Furthermore, our model has also shown remarkable superiority over the TTE-Ensemble model by reducing the errors by (5.19%, 5.5%, and 4.73%) on NYC, Chengdu, and Xi'an, respectively. Thus, we can observe that ST-NN and TTE-Ensemble models achieved better results than ML algorithms (LRM, XGBoost, and LightGBM). This is because deep learning approaches consider the non-linear relations between the variables. Although, the ST-NN applied two DNN modules for estimating the trip distance first, then using this distance to predict the time, which means they also adopted the spatial component (distance) only, while the temporal patterns was ignored. The TTE-Ensemble model was built based on combining the DNN module with the ML (GBDT) model. These models are not sufficient to capture the complicated correlations.

Eventually, as it can be seen from the table, FMA-ETA and STTN models give results which are more closer to our proposed model because these models have also adopted attention mechanisms to capture the non-linear correlations between the spatial and temporal features. The auxiliary spatial features that influence traffic patterns play a significant role when considering the dynamic scales of inner spatial and temporal correlations.

Therefore, compared to FMA-ETA, our proposed model components (SSAM, RDCM, and MHAM) play a significant role in reducing the MAPE and MAE error rates by (2.67%, 3.74%, and 1.91%) and (15.09, 35.42, and 27.25 sec). Also, our model achieves better performance than the STTN model through reducing the errors by at least (1.24%, 2.17%, and 1.61%) and (8.11, 22.68, and 16.78 sec) on MAPE and MAE for NYC, Chengdu, and Xi'an, respectively.

Moreover, to validate our model, two different datasets at morning peak (7 to 10 AM) and evening peak (5 to 8 PM) have been used to test all models during these two periods in terms of MAPE and MAE, as shown in Tables 4 and 5 for NYC, Chengdu, and Xi'an, respectively. Prediction errors are typically higher during these two peak periods than

Table 3: Comparison of all models' results on the NYC, Chengdu, and Xi'an datasets.

Model	NYC		Chengdu		Xi'an	
	MAPE	MAE (sec)	MAPE	MAE (sec)	MAPE	MAE (sec)
LRM	26.12	168.34	24.37	176.33	25.85	197.14
XGBoost	25.39	153.94	22.59	162.19	23.37	188.03
LightGBM	22.19	149.72	21.98	155.39	21.51	176.26
ST-NN	20.04	136.34	19.02	131.26	20.44	154.07
TTE-Ensemble	18.33	122.71	17.58	114.08	18.86	136.35
FMA-ETA	15.81	112.21	15.74	107.17	16.04	121.13
STTNs	14.38	105.23	14.25	94.61	15.74	110.66
JSTC	**13.14**	**97.12**	**12.08**	**71.93**	**14.13**	**93.88**

We denote our model's results in bold font as the best scores for each metric.

Table 4: An illustration of all models' performances with morning and evening peak periods (MAPE) for NYC, Chengdu, and Xi'an.

Model	NYC MAPE		Chengdu MAPE		Xi'an MAPE	
	Morning	Evening	Morning	Evening	Morning	Evening
ST-NN	25.42	26.33	24.12	25.74	26.04	27.16
TTE-Ensemble	22.36	23.65	21.01	23.66	23.75	24.52
FMA-ETA	20.52	21.33	18.77	21.82	20.16	22.93
STTNs	17.78	19.42	16.34	18.25	17.66	19.38
JSTC	**15.82**	**17.13**	**14.52**	**16.04**	**16.27**	**18.45**

We denote our model's results in bold font as the best scores for each metric.

Table 5: An illustration of all models' performances with morning and evening peak periods (MAE) for NYC, Chengdu, and Xi'an.

Model	NYC MAE		Chengdu MAE		Xi'an MAE	
	Morning	Evening	Morning	Evening	Morning	Evening
ST-NN	159.02	164.34	166.62	168.87	163.46	170.19
TTE-Ensemble	145.28	154.11	150.84	155.43	150.22	161.82
FMA-ETA	128.61	133.72	126.96	140.48	137.63	144.57
STTNs	119.83	125.61	117.13	128.72	121.75	134.44
JSTC	**108.74**	**115.93**	**98.31**	**118.49**	**106.59**	**125.16**

We denote our model's results in bold font as the best scores for each metric.

during non-peak periods. From the results shown above, we can demonstrate that our model provides more accurate results compared to other models, even during the morning and evening peak hours. Also, based on the random selection of trips used for testing our proposed model, Figure 7 shows a comparison between actual values and predictions of 50 random trips for all models on NYC, Chengdu, and Xi'an, respectively. Each point on the X-axis represents a trip from the test set, while the y-axis indicates the trip duration in seconds.

4.2.2. Ablation Analysis. We built our model based on three main components (SSAM, RDCM, and MHAM). Besides, we consider external factors that influence travel time by improving the accuracy of our results. Therefore, additional experiments were conducted to verify the contribution of each component in our prediction task. The ablation models we use in this analysis are as follows:

(1) Without SSAM: in this model, we removed the spatial self-attention module (SSAM) and applied RDCN and MHAM modules only with a fully connected and output layer.

(2) Without RDCM: in this model, we removed the DRCM module and applied SSAM and MHAM modules only with a fully connected and output layer.

(3) Without externals: to verify the effect of external factors (weather and public holidays), we remove the block responsible for representing these factors' dependencies.

(4) Without MHAM: we removed a multi-head module (MHAM). So, after getting the spatial and temporal components' correlations, we concatenate these blocks' outputs with external features' representations and then apply fully connected and output layers directly.

We should mention that MLP layers were adopted as an alternative to each module that was removed during the ablation investigation phases 1, 2, and 4, as shown in Table 6. However, the impact of externals was just measured by removing the external factors' representation block in ablation 3. The results in Table 6 demonstrate that the performance of all modules combined together in one model leads to better results. In contrast, removing some parts affects the process of capturing traffic pattern fluctuations.

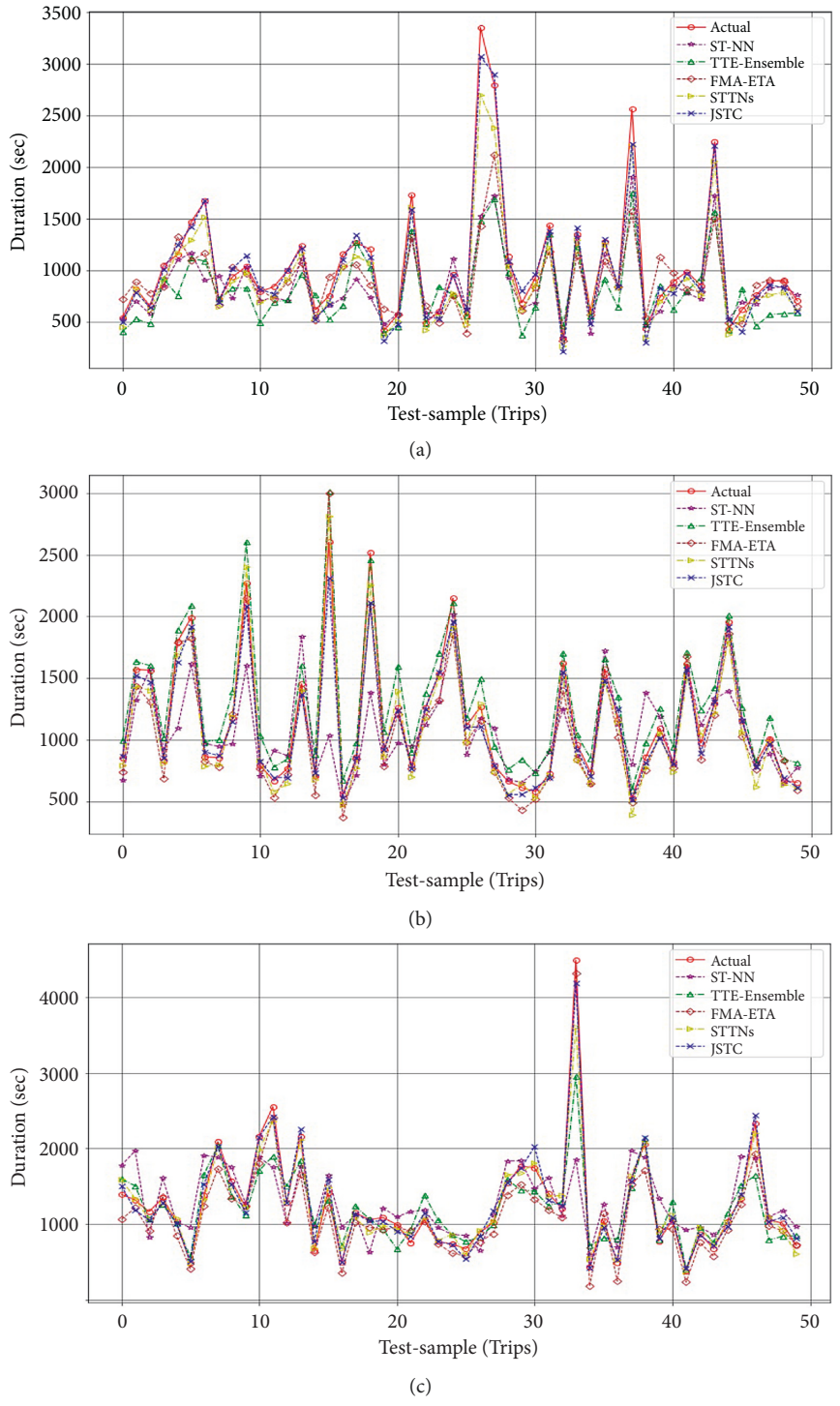

Figure 7: Comparison of prediction vs. actual values for all models on (a) NYC, (b) Chengdu, and (c) Xi'an.

We can notice that the MAPE increases by (8.22, 8.41, and 8.63) and the MAE increases by (55.15, 67.23, and 70.29 sec) in NYC, Chengdu, and Xi'an, respectively, when removing the SSAM block, while removing the RDCM block increases the MAPE by approximately (6.95, 6.25, and 6.28) and MAE by (47.66, 53.98, and 55.5 sec). By removing external factors, the MAPE increases by at least (4.29, 4.85, and 4.92) and the MAE increases by (93.68, 46.44, and 41.66 sec). That means applying external factors improves our model's results by a significant margin. Whereas, applying the SSAM and RDCM modules combined with external factors representation without the MHAM block, we achieve results with small errors rates (MAPE%) about (3.67, 2.87, and 3.45) and MAE at least by (21.42, 38.39, and 27.38 sec) for NYC, Chengdu, and Xi'an, respectively, compared to combining all the JSTC model's components. We can observe that disabling joint

Table 6: Impact of the JSTC model's components: SSAM, RDCM, MHAM, and external factors.

Feature	NYC		Chengdu		Xi'an	
	MAPE	MAE (sec)	MAPE	MAE (sec)	MAPE	MAE (sec)
Without SSAM	21.36	152.27	20.49	139.16	22.76	164.17
Without RDCM	20.09	144.78	18.33	125.91	20.41	149.38
Without externals	17.43	136.82	16.93	118.37	19.05	138.54
Without MHAM	16.81	118.54	14.95	110.32	17.58	121.26
JSTC	**13.14**	**97.12**	**12.08**	**71.93**	**14.13**	**93.88**

We denote our model's results in bold font as the best scores for each metric.

Table 7: Comparison of time consumption of different models on all datasets.

Model	NYC		Chengdu		Xi'an	
	Train (1 epoch) sec	Test (1M trip) sec	Train (1 epoch) sec	Test (1M trip) sec	Train (1 epoch) sec	Test (1M trip) sec
ST-NN	157	122	110	102	114	107
TTE-Ensemble	214	155	119	117	122	115
FMA-ETA	221	218	124	120	137	126
STTNs	244	223	143	127	151	133
JSTC	253	229	168	129	170	136

correlation mechanisms (SSAM and RDCM) increases the error rates more than removing a multi-head block, which means these two modules have a higher impact on our model since they are responsible for capturing correlations of traffic spatial and temporal factors. On the other hand, external factors play an important role in improving our prediction results. Conclusively, these results emphasize the importance of each proposed block through their contributions to improving travel time prediction results.

4.2.3. Computational Cost Measurement. Measuring the computational complexity has been considered in this paper. We compute the time consumption of our model compared with deep learning-based models (ST-NN, TTE-Ensemble, FMA-ETA, and STTNs). Table 7 reports the average time of training and predicting functions for one million trips (1M) with only one epoch on NYC, Chengdu, and Xi'an datasets. Note that we performed our experiments on the same NVIDIA GPU (GeForce GTX 1050 Ti) with 4 GB. Also, we set the batch size to 512 for all models' training phase. Thus, we could observe that the complicated model's structure took more training time than the simple ones. Actually, one logical reason is that this model's complexity represents an improvement to give more accurate prediction results. In comparison, we can notice that the computation time of our model is much closer to that of the STTN model due to the fact that both models have a relevant structure.

5. Conclusion

In this paper, we first discussed the various characteristics of traffic patterns that affect travel time. Then, we presented a mechanism for capturing interactions between spatial and temporal factors based on self-convolutional attention and dilated convolutional techniques. In addition, we adopted spatial auxiliary features and integrated them with the temporal features, which play a significant role in capturing the dynamic traffic patterns and their correlations. Furthermore, we applied a multi-head attention mechanism to learn the attentional weights of the spatial, temporal, and external features based on their contribution to the output and speed up the training process. Extensive experiments using three large-scale real-world traffic datasets (NYC, Chengdu, and Xi'an) have shown that our JSTC model outperforms prior methods.

Data Availability

The terms of use of the data used in this study do not allow the authors to distribute or publish these datasets directly. However, data can be obtained directly from the following webpages: NYC Taxi and Limousine Commission (TLC)—https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page; DiDi Chengdu taxi dataset—https://outreach.didichuxing.com/app-vue/dataList.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This study was supported by the National Key Research and Development Program of China (no. 2021ZD0112400) and the National Natural Science Foundation of China (no. U1811463).

References

[1] F. H. Tseng, J. H. Hsueh, C. W. Tseng, Y. T. Yang, H. C. Chao, and L. D. Chou, "Congestion prediction with big data for real-time highway traffic," *IEEE Access*, vol. 6, Article ID 57311, 2019.

[2] F. Fan, S. Li, W. Dou, and S. Yu, "An evolutionary approach for short-term traffic flow forecasting service in intelligent transportation system," in *Smart Computing and Communication*, M. Qiu, Ed., vol. 10135, pp. 477–486, SmartCom, Springer, Berlin, Germany, 2017.

[3] S. Xu, R. Zhang, W. Cheng, and J. Xu, "MTLM: A Multi-Task Learning Model for Travel Time Estimation," *GeoInformatica*, Springer, Berlin, Germany, 2020.

[4] J. Kwon, B. Coifman, and P. Bickel, "Day-to-day travel-time trends and travel-time prediction from loop-detector data," *Transportation Research Record*, vol. 1717, pp. 120–129, 2000.

[5] X. Zhang and A. John, "Short-term travel time prediction," *Transportation Research Part C: Emerging Technologies*, vol. 11, no. 3-4, pp. 187–210, 2003.

[6] M. Chen and S. I. J. Chien, "Dynamic freeway travel-time prediction with probe vehicle data," *Transportation Research Record*, vol. 1768, no. 01, pp. 157–161, 2001, https://trrjournalonline.trb.org/doi/pdf/10.3141/1768-19.

[7] H. Ji, A. Xu, X. Sui, and L. Li, "The applied research of kalman in the dynamic travel time prediction," in *Proceedings of the 2010 18th International Conference on Geoinformatics*, Beijing, China, June 2010.

[8] T. Oda, "An algorithm for prediction of travel time using vehicle sensor data," *Computer Science*, vol. 40–44, 1990.

[9] D. J. Sun and Z.-R. Peng, "Route travel time estimation based on seasonal model and Kalman filtering algorithm," *Journal of Chang'an University (Natural Science Edition)*, vol. 441, 2014.

[10] J. Xia, M. Chen, and W. Huang, "A multistep corridor travel-time prediction method using presence-type vehicle detector data," *Journal of Intelligent Transportation Systems*, vol. 15, no. 2, pp. 104–113, 2011.

[11] B. Gupta, S. Awasthi, and R. Gupta, "Taxi travel time prediction using ensemble-based random forest and gradient boosting model," *Advances in Intelligent Systems and Computing*, vol. 63–78, 2018.

[12] J. W. C. Van Lint, S. P. Hoogendoorn, and H. J. van Zuylen, "Accurate freeway travel time prediction with state-space neural networks under missing data," *Transportation Research Part C: Emerging Technologies*, vol. 13, no. 5-6, pp. 347–369, 2005.

[13] D. Park and L. R. Rilett, "Forecasting freeway link travel times with a multilayer feedforward neural network," *Computer-Aided Civil and Infrastructure Engineering*, vol. 14, no. 5, pp. 357–367, 1999.

[14] N. Wisitpongphan, W. Jitsakul, and D. Jieamumporn, "Travel time prediction using multi-layer feed forward artificial neural network," in *Proceedings of the - 2012 4th International Conference on Computational Intelligence, Communication Systems and Networks*, pp. 326–330, Phuket, Thailand, July 2012.

[15] Y. Duan, Y. Lv, and F. Y. Wang, "Travel time prediction with LSTM neural network," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 1053–1058, 2016.

[16] Y. Liu, Y. Wang, X. Yang, and L. Zhang, "Short-term travel time prediction by deep learning: a comparison of different LSTM-DNN models," in *Proceedings of the IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 1–8, Yokohama, Japan, October 2017.

[17] D. Wang, J. Zhang, W. Cao, J. Li, and Y. Zheng, "When will you arrive? Estimating travel time based on deep neural networks," in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pp. 2500–2507, New York, NY, USA, Febraury 2018.

[18] Y. Sun, Y. Wang, and K. Fu, "FMA-ETA: estimating travel time entirely based on FFN with attention," 2020, https://arxiv.org/abs/2006.04077, Article ID 04077.

[19] J. Ishan and Q. Tony, "A unified neural network approach for estimating travel time and distance for a taxi trip," in ", http://arxiv.org/abs/1710.04350, 2017.

[20] Y. Li, C. Shahabi, and K. Fu, "Multi-task representation learning for travel time estimation," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1695–1704, London UK, August 2018.

[21] Z. Zou, H. Yang, and A. Xing Zhu, "Estimation of travel time based on ensemble method with multi-modality perspective urban big data," *IEEE Access*, vol. 8, no. 2, Article ID 24819, 2020.

[22] Z. Jia, C. Chen, B. Coifman, and P. Varaiya, "The PeMS algorithms for accurate, real-time estimates of g-factors and speeds from single-loop detectors," in *Proceedings of the IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 536–541, Oakland, CA, USA, August 2001.

[23] W. H. Lee, S. S. Tseng, and S. H. Tsai, "A knowledge based real-time travel time prediction system for urban network," *Expert Systems with Applications*, vol. 36, no. 3, pp. 4239–4247, 2009.

[24] D. Billings and J. S. Yang, "Application of the ARIMA models to urban roadway travel time prediction - a case study," in *Proceedings of the Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, vol. 3, pp. 2529–2534, Taipei, Taiwan, October 2006.

[25] B. Daniel, T. Olli Pekka, S. Blandin, A. M. Bayen, T. Iwuchukwu, and K. Tracton, "An Ensemble Kalman Filtering approach to highway traffic estimation using GPS enabled mobile devices," in *Proceedings of the IEEE Conference on Decision and Control*, pp. 5062–5068, Cancun, Mexico, December 2008.

[26] C.-H. Wu, J.-M. Ho, and D. T. Lee, "Travel-time prediction with support vector regression," *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, no. 4, pp. 276–281, 2004.

[27] Y. Zhang and A. Haghani, "A gradient boosting method to improve travel time prediction," *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 308–324, 2015.

[28] B. Yang, C. Guo, and C. S. Jensen, "Travel cost inference from sparse, spatio temporally correlated time series using Markov models," *Proceedings of the VLDB Endowment*, vol. 6, no. 9, pp. 769–780, 2013.

[29] Y. Jing, "T-drive: driving directions based on taxi trajectories," in *Proceedings of the GIS '10: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 99–108, California, USA, November 2010.

[30] M. Rahmani, E. Jenelius, N. Haris, and Koutsopoulos, "Route travel time estimation using low-frequency floating car data," in *Proceedings of the IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, The Hague, Netherlands, October 2013.

[31] M. Rahmani, E. Jenelius, and H. N. Koutsopoulos, "Floating car and camera data fusion for non-parametric route travel time estimation," in *Proceedings of the 17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 1286–1291, Qingdao, China, October 2014.

[32] X. Zhan, S. Hasan, and V. Satish, "Ukkusuri and Camille Kamga. "Urban link travel time estimation using large-scale

taxi data with partial information"," *Transportation Research Part C: Emerging Technologies*, vol. 33, no. 37–49, 2013.

[33] Y. Wang, Y. Zheng, and Y. Xue, "Travel time estimation of a path using sparse trajectories," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 25–34, Newyork, USA, Auguest 2014.

[34] T. Xu, X. Li, and C. Claramunt, "Trip-oriented travel time prediction (TOTTP) with historical vehicle trajectories," *Frontiers of Earth Science*, vol. 12, no. 2, pp. 253–263, 2018.

[35] T.-Y. Fu and W.-C. Lee, "Deepist: deep image-based spatio-temporal network for travel time estimation," pp. 69–78, 2019.

[36] E. Faruk, "Commercial vehicle travel time estimation in urban networks using GPS data from multiple sources," *Computer Science*, vol. 12, pp. 141–151, 2013.

[37] M. Xu, W. Dai, C. Liu et al., "Spatial-temporal transformer networks for traffic flow forecasting," 2020, https://arxiv.org/abs/2001.02908.

[38] G. Liang, S. Li, Y. Wang, F. Chang, and K. Wu, "Global spatial-temporal graph convolutional network for urban traffic speed prediction," *Applied Sciences*, vol. 10, no. 4, 2020.

[39] Tcl, "NYC taxi and limousine commission (TLC)," https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page, 2021.

[40] M. Abdollahi, T. Khaleghi, and K. Yang, "An integrated feature learning approach using deep learning for travel time prediction," *Expert Systems with Applications*, vol. 139, 2020.

[41] H. Peng, H. Wang, B. Du et al., "Spatial temporal incidence dynamic graph neural networks for traffic flow forecasting," *Information Sciences*, vol. 521, pp. 277–290, 2020.

[42] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," 2017, https://arxiv.org/abs/1706.03762, Article ID 03762.

[43] P. Bholowalia and A. Kumar, "Ebk-means: a clustering technique based on elbow method and k-means in wsn," *International Journal of Computer Application*, vol. 105, no. 9, 2014.

[44] L. Nahar, Z. Sultana, and Z. Sultana, "A new travel time prediction method for intelligent transportation system," *IOSR Journal of Computer Engineering*, vol. 16, no. 3, pp. 24–30, 2014.

[45] Kaggle, "Weather data in New York city - 2016 | kaggle," 2016, https://www.kaggle.com/mathijs/weather-data-in-new-york-city-2016.

[46] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, https://arxiv.org/abs/1412.3555.

[47] Y. Wu and H. Tan, "Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework," 2016, https://arxiv.org/abs/1612.01022.

[48] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, and J. Zhang, "Urban traffic prediction from spatio-temporal data using deep meta learning," in *Proceedings of the KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1720–1730, Anchorage AK, USA, July 2019.

[49] N. Ranjan, S. Bhandari, H. P. Zhao, H. Kim, and P. Khan, "City-wide traffic congestion prediction based on cnn, lstm and transpose cnn," *IEEE Access*, vol. 8, Article ID 81606, 2020.

[50] P.-Y. Ting, T. Wada, Y.-L. Chiu, M. T. Sun, K. Sakai, and W. S. Ku, "Freeway travel time prediction using deep hybrid model-taking sun yat-sen freeway as an example," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 8257–8266, 2020.

[51] K. Guolin, M. Qi, T. Finley et al., "LightGBM: a highly efficient gradient boosting decision tree," in *Proceedings of the NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, December 2017.