IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 26, 2020, accepted April 10, 2020, date of publication April 22, 2020, date of current version May 7, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2989443

Spatiotemporal Data Fusion in Graph Convolutional Networks for Traffic Prediction

BAOXIN ZHAO[1,2], (Student Member, IEEE), XITONG GAO[2], (Member, IEEE), JIANQI LIU[3], (Member, IEEE), JUANJUAN ZHAO[2], AND CHENGZHONG XU[4], (Fellow, IEEE)

[1]Shenzhen College of Advanced Technology, University of Chinese Academy of Sciences, Shenzhen 518055, China
[2]Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China
[3]School of Automation, Guangdong University of Technology, Guangzhou 510006, China
[4]State Key Laboratory of IoTSC, Faculty of Science and Technology, University of Macau, Macao, China

Corresponding author: Xitong Gao (xt.gao@siat.ac.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2019YFB2102100, in part by the National Natural Science Foundation of China under Grant 61701122, Grant 61806192, and Grant 61802387, in part by the Science and Technology Development Fund of Macao (FDCT) under Grant 0015/2019/AKP, in part by the Shenzhen Engineering Research Center for Beidou Positioning Service Improvement Technology under Grant XMHT20190101035, in part by the Shenzhen Discipline Construction Project for Urban Computing and Data Intelligence, in part by the Basic Research Program of Shenzhen under Grant JCYJ20180302145731531 and Grant JCYJ20190812160003719, and in part by the Research Center for Ecology and Environment of Central Asia, Chinese Academy of Sciences.

ABSTRACT A plethora of information is now readily available for traffic prediction, making an effective use of them enables better traffic planning. With data coming from multiple sources, and their features spanning spatial and temporal dimensions, there is an increasing demand to exploit them for accurate traffic prediction. Existing methods, however, do not provide a solution for this, as they tend to require expertise feature engineering. In this paper, we propose a general architecture for **S**patio**T**emporal **D**ata **F**usion (STDF) with parameter efficiency. To make heterogeneous multi-source data fusion effectiveness, we separate all data into traffic directly related data and traffic indirectly related data. With traffic indirectly related data as the input to **S**patial **E**mbedding by **T**emporal convoluti**ON** (SETON) that simultaneously encodes each feature in both space and time dimensions and traffic directly related data as the input to the graph convolutional network(GCN), we designed a fine-grained feature transformer to match the ones generated by GCN. This is then followed by a fusion module to combine all features to make final prediction. Compared to using GCNs training with only traffic directly related data, experimental results show that our model can achieve a 6.1% improvement in prediction accuracy measured by Root Mean Squared Error.

INDEX TERMS Data fusion, graph convolutional networks, multi-source data, traffic prediction.

I. INTRODUCTION

The traffic is playing a vital role of human life and significantly influenced every aspect of life. With the rapid increase of vehicles, the traffic jam has attracted a national concern for urban management. Smart city is considered as a potential solution, which uses intelligent technologies to predict the traffic flow, and smooth the peaks and valleys by offering residents travel guidance [1]. Traffic prediction is of great importance for smart cities and has attracted attention from research to industry for many years. Accurate real-time road traffic prediction is critical for the realization of intelligent cities [2], [3]. Traffic authorities require reliable prediction to facilitate the related process of policy-making, regulatory, and implementation. With the development of sensors, the traffic data is collected by sensors equipped within vehicles or installed along the roads. Examples of traffic data include license number of vehicles, GPS data of vehicles, video or image records of surveillance devices, temperature, wind speed and level of sunlight data of weather sensors [4]. These multi-source data converge to the data center by vehicle ad hoc networks (VANET), or the upcoming

The associate editor coordinating the review of this manuscript and approving it for publication was Keli Xiao.

5G cellular network [5]. Many traffic prediction algorithms have proposed to guide convenient travel for citizens based on these mass traffic data, and there have been some works showing the advantages of multi-source data fusion in the spatio-temporal data prediction tasks [6].

Unlike traditional data fusion methods, multi-source traffic data includes not only traffic directly related data, *e.g.*, vehicle speed, vehicle density, traffic flow, but also indirectly related data, *e.g.*, weather, points of interests (PoIs), *etc.*. All these data span both spatial and temporal dimensions [4], [7]. As shown in Figure 1, our goal is to predict traffic condition at each road segment. The residence areas in the morning tend to have many people going out for fun or work, while in the evening many people go home, or to a place of entertainment. These information therefore must be incorporated in the model for accurate traffic prediction. However, merging all these data straightforwardly could not explore the semantics changing of traffic indirectly related data over time. To tackle this, in this paper we propose a **S**patio**T**emporal **D**ata **F**usion (STDF) framework, which is a general architecture to improve traffic prediction performance in metro-city scales by using data fusion.

Traffic prediction using data fusion needs to consider both spatial and temporal features from multi-source data. It is notable that the distribution of urban traffic exhibit high variability both in spatial and temporal domain. Traffic prediction in urban cities is challenging because of their complex environment. It is thus essential to find an efficient and effective way to make traffic prediction more accurate by using them jointly. There have been many works on data fusion. According to the model parameters size, the work in traffic prediction by data fusion can be classified into two categories, *i.e.*, traditional machine learning method and deep neural networks. Many effective methods have been proposed, such as XGBOOST [8], random forest [9], LightGBM [10], embedding learning [11]. Although these methods can find the relationship between traffic prediction and traffic indirectly related data, they require significant human effort because the features extracted from multi-source data play a vital role in the prediction accuracy. Meanwhile, it is computation consuming if we apply these methods into large scale data fusion for urban cities. To overcome it, an end-to-end learning method is thus a desirable alternative at the cost of computation power. For example, some works use deep neural networks [12]–[14] to automate the processing of multi-source data fusion and extraction of useful features. They merge multi-source data straightforwardly into a vector and treat traffic directly related and indirectly related data equally. As a result, they ignore the semantics changing of traffic indirectly related data. In this paper, we explore an effective and efficient way to fuse multi-domain data considering both the spatial and temporal properties based on the GCN.

Multi-source data fusion with the consideration of its spatial and temporal properties is challenging for the following reasons. The first challenge is the large scale feature representation. It is infeasible to encode each node at different time into a unified vector in metro-city scales. For example, the parameter size is over 10G for a Small city containing 10,000 road segment and 100 external factors for each node on average if each factor is represented by a 10-tuple vector at one time interval, which will easily result in an over-parameterized model and over-fitting when training. Second, an automated but efficient facility is urgently needed to find the spatio-temporal representation for all multi-source data. Third, fusing traffic indirectly related data into traffic prediction may cause negative effect on prediction accuracy. Besides, there are practical concern when applied into the real traffic prediction in metro city scales.

To tackle the aforementioned challenges, we proposed a general STDF framework. STDF adopts branching-transfer-fuse strategy. STDF first separates the prediction model into two branches with each branch processing traffic directly related data and traffic indirectly related data correspondingly. The traffic directly realted data is processed by GCN to get the spatio-temporal representation from the middle layer of GCN. While the traffic indirectly related data is process by two parts successively. The first part is called static **S**patial **E**mbedding by **T**emporal convoluti**ON** (SETON). SETON first encodes each feature in both space and time dimensions simultaneously, followed by an convolutional operation with spatial embeddings as input and temporal embeddings as convolutional kernel to get the spatio-temporal representation. Meanwhile, all nodes share the same spatial and temporal embeddings, which are traninable in the model as well as to avoid the overparameterized problem. The second part is a feature transform module which is to map the spatio-temporal representation generated by SETON to the feature map space generated by GCN. At last, the feature map generated by GCN and feature transform module are fused together followed by several full connection output layers. In summary, this paper has the following contributions.

- Generic Architectures for Deep Spatio-temporal Data Fusion - The STDF framework is a general neural network architectures, which can efficiently fuse multi-source data both in spatial domain and temporal domain in large scales.
- Deep Spatio-temporal Data Fusion Operator- We designed a new type of deep spatio-temporal data fusion operator *i.e.*SETON. The operator has the ability to capture both the spatial representation and temporal representation simultaneously.
- Computation Efficiency and Practical - Both the components in the STDF framework have the parameter sharing strategy to avoid model over-parameterized, which is applicable in the complex urban computing with high computation efficiency.
- Performance Improvement in Spatiotemporal Data Prediction - We apply our method into real traffic speed prediction and human flow prediction in metro. Experimental results demonstrate that our spatiotemporal data

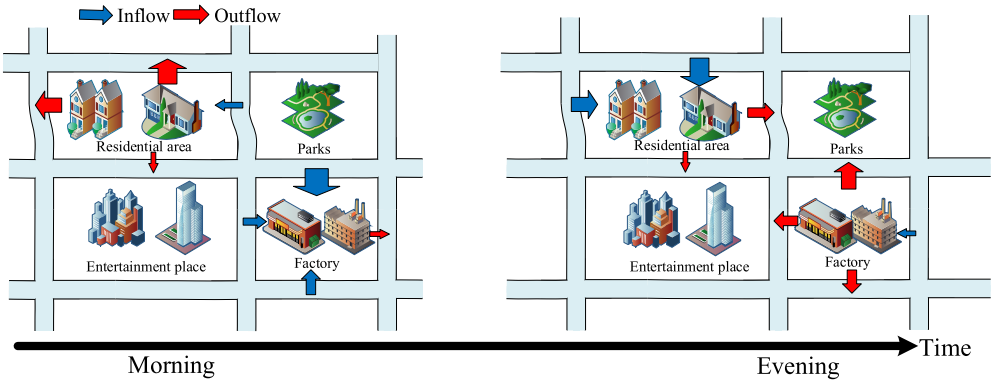


FIGURE 1. Semantics changing over time. Multi-source data fusion will benefit the traffic prediction accuracy. At the same time, the representation of different factors is changing over time. Traffic prediction using data fusion needs to consider both the spatial and temporal representation simultaneously.

fusion method performs significantly better than the one without data fusion or only spatial data fusion.

The rest of this paper is organized as follows. Section II gives a brief literature review of related work from traffic prediction and data fusion perspective. Section III formulates the traffic prediction problem and an overview of the architecture of solution. Section IV details the process of spatiotemporal representation with parameter efficiency. With the extracted features, a feature transformer module and data fusion method are introduced at Section V. We conduct comprehensive experiments in section VI and give a discussion of our model. Section VII offers the conclusion of our work and outlines our future work.

II. RELATED WORK

In this section, we review the recent studies that are relevant to traffic prediction and data fusion. We first introduce the traffic prediction methods from mathematical model perspective. Then data fusion methods are detailed both in feature level and semantic level.

A. TRAFFIC PREDICTION

There are many achievements made in traffic prediction, including traffic flow, vehicle speed, vehicle density, etc. Traffic prediction can be models as a time series data prediction. The statistical modes including history average (HA), Autoregressive Integrated Moving Average (ARIMA) [15], Seasonal Autoregressive Integrated Moving Average (SARIMA) [16] and spatiotemporal correlations [17] are widely used in real traffic condition prediction for its computation efficiency. However, all these methods require the input data to meet a certain condition, which consequently perform poorly in the complex urban traffic prediction.

To make traffic prediction model have the ability to deal with complex data, there are continuous applying trying machine learning methods into the urban traffic prediction, such as XGBOOST [8], random forest [9], LightGBM [10], embedding learning [11]. Although these methods have the inherent advantage to deal with multi-source data, they need a lot of domain knowledge and careful feature engineering, which is not only computation consuming but also has some scalability issues.

Because of the strong self-adapting and self-learning ability of artificial neural network, deep learning has been used in different domains, such as computer vision [18], natural language processing and auto driving, and brings many significant breakthroughs. At the same time, a great deal of studies have been done on improving traffic prediction performance by using different types of neural network architectures, such as multi-layers perception [19], long short-term memory [20] and auto encoders [21]. Although these works can effectively extract the local patterns of data, they can only be applied for the standard structure data and are lack of awareness of the global prediction. With the ability of processing data of graph structures, the graph convolutional networks are widely used to deal with complex graph data in a global perspective. Yu proposed a Spatio-Temporal Graph Convolutional Networks (STGCN) with the ability to capture comprehensive spatial and temporal dependencies for long-term traffic prediction [22]. Guo applies attention strategy into GCN to predict traffic flow with considering the dynamic spatial-temporal correlations of traffic data [23]. Li proposed a diffusion convolutional recurrent neural network (DCRNN) to model the traffic flow as a diffusion process [24]. Different from our work, these models did not deal with the multi-source data problem.

B. DATA FUSION

Data fusion [25] in traffic scenario often implies the combination of traffic related data sets that present an enormous diversity on the basis of location, weather, points of interests, traffic flow, density and speed. These data sets are differently represented in different perspective, but they represent the same real world object and complement each other. A straightforward method [26], [27] in the feature level is that all the object-related features are extracted equally and all features are concatenated sequentially into a equal-sized

or unequal-sized vector to be injected into the kernel task. The low-level representation might exist redundancies and the sampled data may be not independent, it is easy to lead to model instability.

Feature engineering is an especially good idea that makes machine learning algorithms work. Lakhinaet analyzed the distributions of packet features in flow traces in details, which showed significant advantages for anomalies detection [28]. Samant and Adeli extracted traffic incident related features by using wavelet transform and linear discriminant analysis [29]. The two-stage feature extraction algorithm made the traffic incidents detection model more robust. Although a good feature engineering can get better performance, it needs a deep understanding of domain knowledge. Besides, it is time consuming and computation consuming for large scale data fusion. An end-to-end learning technology with better flexibility provides a consistent alternative for the ability of auto feature extraction.

Deep neural networks (DNN) is an excellent solution for end-to-end learning when geta unified feature representation from disparate data sets. An end-to-end structure of ST-ResNet [12] was proposed to predict citywide crowd flows, where the input with unique properties of spatiotemporal data is feed into ST-ResNet simultaneously. Bojarski trained a convolutional neural network (CNN) to map raw pixels from three cameras directly to steering commands [30]. The system automatically learns internal representations of the necessary processing steps such as detecting useful road features. With the ability to self-learn feature representation, these end-to-end based data fusion methods need lots of computation cost. At the same time, the feature representations are extracted in a grid scale, but not in the road segments level. Different from them, we are more interested in the graph structure data.

Feature based data fusion approaches take all the feature equally and ignore the semantic meaning of each feature. On the contrary, semantics based data fusion methods try to understand the meaning of each feature and find the relationships between features by mining the insight of each data. For example, many works tried to find the relationship between emotion and audio signals in the emotion recognition [31]–[33]. The fusion results combining the acoustic and facial emotion recognition were achieved in the semantic level. DeepFM [34] is an end-to-end deep learning framework for click-through rate prediction, where data representation is realized by feature embedding. DeepFM fuses the feature by a factorization-machine with a deep neural network. However, all these feature representation are static and only related to its input data correspondingly. In this paper, we will tackle the spatiotemporal data fusion problem in traffic prediction scenarios because the spatial features in semantics level are dynamically changing with time.

III. PRELIMINARIES

Definition 1: (Spatial Network): The traffic network G is a weighted directed graph $G = (V, E, A)$, where set $|V| = N$ is a set of nodes that can represent road segments or metro stations, N is the number of nodes, and E denotes the set of edges, $A \in \mathbb{R}^{N\times N}$ is the weighted adjacent matrix of network G.

Definition 2: (Multi-Source Data): The multi-source data includes two types of data, traffic directly related data and traffic indirectly related data. The traffic directly related data means the graph signal matrix $X_G^t \in \mathbb{R}^{N\times C}$, where C is the number of traffic condition of interests (e.g., traffic speed, traffic flow, traffic density, etc.). The traffic indirectly related data represents external factors that can influence traffic condition indirectly, which is denoted by $F_G \in \mathbb{R}^{N\times M}$, where M is the number of fields including categorical fields (e.g., residential area, hi-tech zones, entertainment place, rain, etc.) and continuous fields (e.g., PoI density, PoI number).

A. PROBLEM STUDIED

The problem of traffic prediction by data fusion can be described as: given the observations at N nodes of historical P time steps $\mathcal{X} = (X_G^{t-P+1}, X_G^{t-P+2}, \cdots, X_G^t) \in \mathbb{R}^{P\times N\times C}$ and the external factors F_G collected from other domain, we aim to learn a mapping function f which can map the input data into the future observation of traffic condition $Y = (X_G^{t+1}, X_G^{t+2}, \cdots, X_G^{t+Q})$, i.e., $Y = f(\mathcal{X}, F_G)$, where Q denotes the length of the target of traffic condition to predict.

Figure 2 illustrates the architecture of STDF framework to solve the problem. As the studies about GCN have gotten state-of-the-art performance in spatio-temporal data prediction and there have been many completed GCN architectures widely used in time series data prediction, we select one type of GCN [22] to demonstrate the framework of STDF.

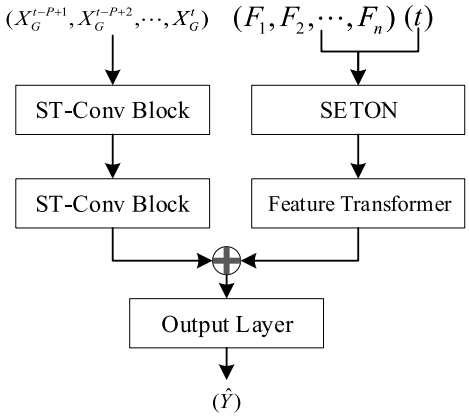


FIGURE 2. STDF architecture. The SETON operator aims to getting the low-level spatio-temporal features from the input including spatial features and temporal factors. The feature transformer component is to map the low-level features to a high level feature that has the same size with the high-level features generated by GCN. The ST-Conv Block is the basic block for GCN.

B. GRAPH CONVOLUTIONAL NETWORK

Graph convolutional network (GCN) is a neural network that operates on graphs, which is able to extract local features with different reception fields from translation variant

non-Euclidean structure [35]. As depicted in [22], GCN is designed to solve the time-series prediction problem, i.e., predicting the future traffic measurements under given input with a fixed temporal length, which is written as

$$\widehat{Y} = GCN(X_G^{t-P+1}, X_G^{t-P+2}, \cdots, X_G^t).$$

The feature map FM^g generated by the second ST-Conv Block in GCN as demonstrated in Figure 2 is denoted by $FM^g = f_g(X_G^{t-P+1}, S_G^{t-P+2}, \cdots, S_G^t)$.

However, there are many external factors that have influence on traffic pattern. For each node v, the external factors are written as a vector F_v. Spatio-temporal data fusion is not a simple data integration process. STDF is designed to find an efficient and effective data fusion strategy that is one kind of practical methods for large scale traffic data prediction in real world. STDF consists of three parts: SETON and Feature Matching. The SETON is to find a computation efficient spatio-temporal representation for external traffic related variables. Feature transformer maps spatio-temporal representation to a feature space that has the same feature shape with the feature map FM^g for each node. Then a fusion module is followed to combine the two features into one tensor. We introduce the three parts in details as follows.

IV. SETON

The STEON consists of three components: spatial feature embedding layer, temporal feature embedding layer and embedding vector fusion layer. The spatial feature embedding layer maps the external factors to a fixed sized embedding vector. The vector length k is determined in advance. Similarly, the temporal feature embedding layer maps the time interval to a 3-D tensor, with the length of the first and second dimension equal to k and the length of the third dimension equal to the number of time slots, and embedding vector fusion layer is to get the spatio-temporal embedding vectors using the output of the aforementioned two components as input, which is the spatio-temporal representation of traffic indirect related data in low level. At the same time, all vectors in SETON can be self-learned without any feature engineering.

A. SPATIAL FEATURE EMBEDDING

Because the traffic network is complex and the environment around each node is different from each other, the size of external data related to traffic prediction is too large if we give each factor a spatiotemporal representation in neural networks, which may cause over-parameterized and overfitting when training. To overcome the over-parameterized problem, we proposed a data sharing strategy for all nodes.

We first classify the indirect traffic data into an m-fields data according to the way how the PoI will influence people travel pattern. They may include categorical fields (e.g., residential area, hi-tech zones, entertainment place) and continuous fields (e.g., PoI density, PoI number). Different categorical fields may contains different size of data denoted by an one hot encoding. The continuous fields are represented

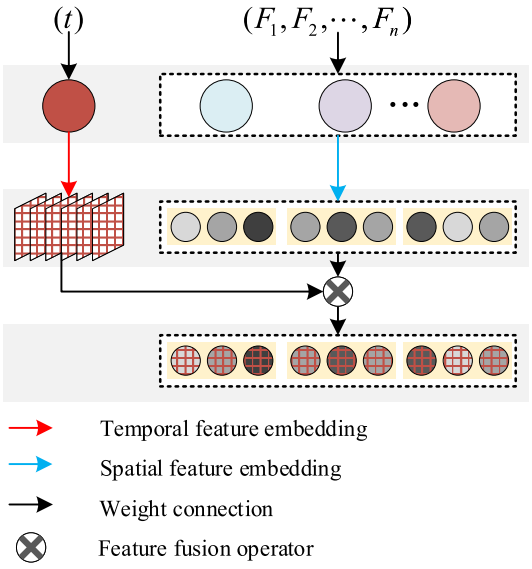


FIGURE 3. SETON architecture. The STEON consists of spatial feature embedding layer, temporal feature embedding layer and spatiotemporal feature product layer. SETON is designed to get the feature representation of indirect traffic data both in spatial and temporal dimension simultaneously with parameter efficiency.

by the value itself. The instance for node v is written as $F_v = \{f_{field_1}, f_{field_2}, \cdots, f_{field_m}\}$, where f_{field_j} stands for the j-th field of F_v. Then the instance for all node V is $F = \{F_1, F_2, \cdots, F_n\}$. The task for spatial feature embedding is to find a parameter efficient method to allocate each value in F to a equal sized embedding vector. The length of embedding vector is a predefined as k.

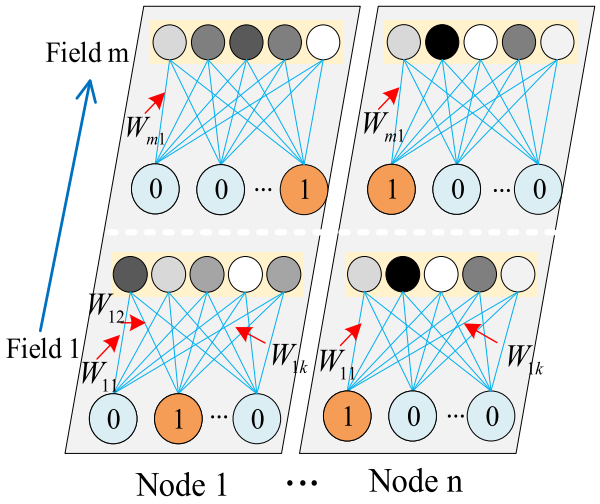


FIGURE 4. Spatial feature embedding. All nodes share a same latent feature vectors W. The tensor W serves as network weights which are trainable and acts as a role in mapping the input data to fixed sized embedding vectors, which makes our model salable when appled into large urban computing.

Figure 4 highlights the detail of spatial feature embedding from the input layer F to the embedding layer for all nodes,

where the length of embedding vector is set to 5. The left part stands for the embedding process for node 1 and the right part stands for the same process for node n. All nodes share a same latent feature vectors W. By the way, there is no need of pre-training for the latent feature vectors W. The tensor W serves as network weights, which can be learned by the network itself. Besides, the tensor W acts as a role in mapping the input data to fixed sized embedding vectors, which denoted as:

$$a_i = [e_{i,1}, e_{i,2}, \cdots, e_{i,m}],$$

where $e_{i,j}$ stands for the embedding vector of j-th field for node i and m is the number of fields. More specifically, the embedding output for each node is a $k \times m$ tensor. The parameter that needs to be learned is of a size of $M \times k$, where M is equal to $\sum_{j=1}^{m} |f_{field_j}|$. The parameter size has no relationship with the node number, which is the foundation for large scale data fusion in urban cities.

B. TEMPORAL FEATURE EMBEDDING

In the application of traffic prediction, time factor plays an important role in understanding people travel patterns [7]. For example, people would like to go out in the morning and get back home in the evening. So the embedding vector for residential area is different at different time. Meanwhile, the spatial semantics changing is also needed for other kinds of categories. Because the characteristics of traffic data has a property of cyclical, we divide the time in one day into T time intervals. As we all know, urban traffic has different patterns and people travel patterns also differs from each other at different time. So each time interval has a distinct transmission matrix in our model, which is used to transfer the spatial embedding vector to a corresponding vector.

We use a tensor $\mathcal{W}$ to represent the temporal embeddings for all time intervals. At the same time, $\mathcal{W}$ serves as network weights which can be learned by the network itself. To make the temporal embeddings matching with the spatial embeddings, the tensor $\mathcal{W}$ is of a 3-D shape $T \times k \times k$.

Taking $k = 5$ as an example, we highlight the temporal feature embedding process from the input layer to the embedding layer as shown in Figure 5. For the time factor t, we encode it to a one hot vector after discretization, where the vector length is equal to T. Similarly, the latent feature vectors $\mathcal{W}$ for temporal embedding process serves as network weights which can be learned by the network itself. After the embedding process, we get the temporal embedding matrix α_t corresponding to the time interval t.

$$\alpha_t = f(\mathcal{W}, t) = \mathcal{W}[t, :, :],$$

where f is a look up function to get its corresponding vector. And the output matrix α_t has a shape of $k \times k$, which stands for how the spatial meaning of each categories changes over its corresponding time t. The size of parameters $\mathcal{W}$ has relationship only with time intervals and embedding size but not determined by node number n, which benefits large scale data fusion in urban cities. Therefore, the spatial embeddings and temporal embeddings make our model scalable without the influence from graph size.

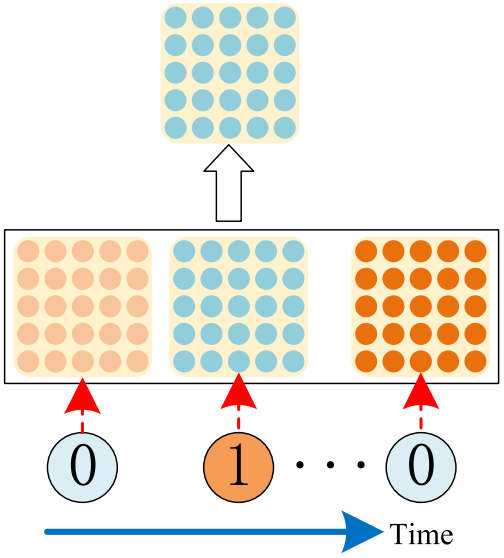


FIGURE 5. Temporal feature embedding. The temporal embeddings serves as network weights with a size of $T \times k \times k$. Each time interval t has its own temporal embedding.

C. SPATIO-TEMPORAL FEATURE REPRESENTATION IN LOW LEVEL

To get both the spatial and temporal feature representation, we apply temporal embedding matrix α_t to every field of spatial embedding vector for all nodes. For a node i at time t, its spatiotemporal feature embedding vectors is calculated by:

$$\begin{aligned} a_i^t &= [\alpha_t * e_{i,1},\ \alpha_t * e_{i,2},\ \cdots,\ \alpha_t * e_{i,m}] \\ &= [e_{i,1}^t, e_{i,2}^t, \cdots, e_{i,m}^t], \end{aligned} \tag{1}$$

where $*$ means the convolutional operator.

In summary, the number of parameters learned by the network itself is only $M \times k + T \times k \times k$. And the output spatiotemporal feature embedding vectors A_t^0 after SETON operation is of a shape of $n \times k \times m$. As similar to the concept in convolutional neural network for computer vision task, this feature representation is in low level.

V. FEATURE TRANSFORMER AND DATA FUSION

This section introduces a feature transformer method that is to get the representation in high level and match the feature map FM^g calculated by GCN.

A. EXTRACT SPATIOTEMPORAL REPRESENTATION IN HIGH LEVEL

As illustrated in Figure 2, the feature transformer component is a bridge between SETON and the feature map FM_g, which achieves shape alignment between the two layers. The feature transformer part stacks q convolutional layers. Each convolutional layer contains a 1-D convolutional kernel which enables all nodes in graph G share the same convolutional

kernel, rectified linear units and batch normalization except the last layer containing only convolutional operation.

$$A_t^l = bn(relu(conv(A_t^{l-1}, k^l))),$$

where k^l is a 1-D vector that can be learned by network itself, $l \in \{1, 2, \cdots, q\}$ stands for the layer number. All nodes share the same convolutional kernel k^l at layer l, which not only makes parameters efficient but also avoid overfitting when training. What's more, the padding operation, incidentally, depends on whether up-sampling is necessary. For example, the feature map $FM^g \in \mathbb{R}^{n \times k_g \times c_g}$ with c_g channels generated by GCN is regarded as a representation for the objective traffic data in high level. If $k < k_g$, up-sampling is necessary and experimental results tell us will cause performance degradation sharply. So it is better to keep the value of k_g is less than k. After the node-wised convolutional operation, the output feature map A_t^q is denoted by FM^{st}, which has the same size with FM^g.

B. FEATURE MAP FUSION

There are many feature map fusion methods widely used in neural networks. But in the large urban cities computing, we prefer to directly merge the feature map FM^{st} generated by STDF with that of GCN as shown in Figure 2, which is denoted by FM followed by rectified linear units and batch normalization and written as:

$$FM = FM^g + FM^{st}.$$

This type of feature map fusion method has two benefits used in large scale data fusion. The first is to reduce the computation overload when add more data into traffic prediction. Besides it would not bring more parameters into our model, thus it can avoid overfitting problem.

To get the predicted value, several full connected layers are stacked to map the feature map to the object value.

C. LOSS FUNCTION

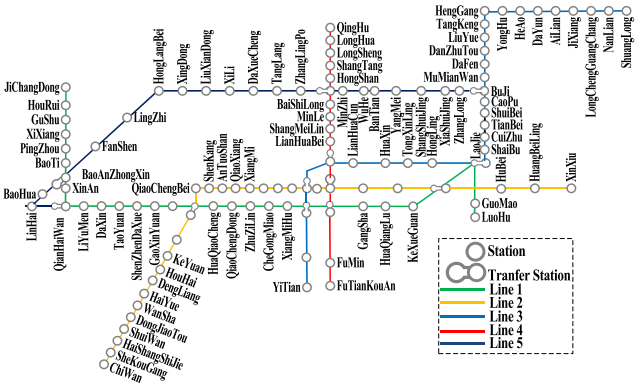
In the training process, the goal is to minimize the gap between the real traffic condition$Y$ and the predicted value$\widehat{Y}$. Different from other tasks, traffic prediction has data incomplete and data bias problem. In statistics, the Huber loss is a loss function used in robust regression, that is less sensitive to outliers in data than the squared error loss. To minimize the influence of traffic outliers, we select Huber loss as the loss function.

$$L(Y, \widehat{Y}) = \begin{cases} \frac{1}{2}(Y - \widehat{Y})^2 & for\ |Y - \widehat{Y}| \le \delta \\ \delta|Y - \widehat{Y}| & otherwise. \end{cases} \quad (2)$$

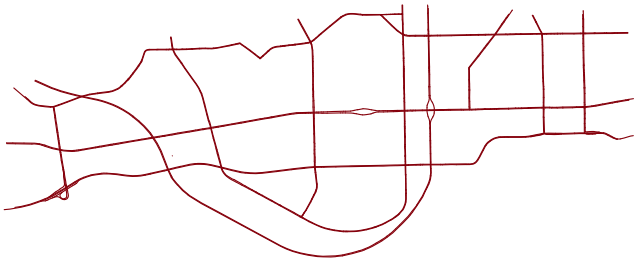
where δ is a threshold parameter which controls the range of squared error loss.

VI. EXPERIMENTS

In this section, we present the experiment and comparison results. We first present the experiment settings with baseline algorithms and datasets introduced, then demonstrate the overall performance of STDF with its components analysis. Finally we detail the training process, testing performance and hyperparameters selection.



(a) Metro



(b) TaxiSZ

FIGURE 6. Physical map in Shenzhen: (a) metro station; (b) road network.

A. EXPERIMENT SETUPS

1) DATESETS

- *Metro:* The dataset used in this study is the smart card transaction records and train operation logs in Shenzhen, China. The metro system has 5 metro lines by 2015 as shown in Figure 6(a). The whole data collected from around 4 million smart cards have more than 300 million smart card transaction records, covering 184 consecutive days from January 1, 2015 to July 30, 2015. We use 144 days of data to train the network and 20 days for cross validation and 20 days for testing. The standard time interval is set to 30-minutes. The prediction task for Metro is to forecast the passenger number at each metro station.
- *TaxiSZ:* We collect the data from City Traffic Bureau of Shenzhen (China) for as long as one year from January 1, 2015 to December 31, 2015. There are about 15,000 taxis equipped with high-resolution GPS devices reporting 32,205,000 records per day on average. There are 341 days of valid data, where 281 days of data is used for training, 30 days for cross validation and 30 days for testing. The standard time interval is set to 15 minutes. We use this data to predict the traffic speed at every road segment as shown in Figure 6(b).

TABLE 1. Statistics on datasets.

Datasets	# Metro	# TaxiSZ
Data Type	Smart Card	GPS
Time Span	1/1/2015 - 7/1/2015	1/1/2015 - 12/1/2015
Valid Data	184 days	341 days
Time Interval	30 minutes	15 minutes
Network Size	118	1376
Sampling Rate	/	20s
Prediction Task	Passenger Number	Traffic Speed
Traffic Indirect Related Data		
Weather Condition	16 types	16 types
PoIs	20 primary categories 135 secondary categories	20 primary categories 135 secondary categories

- *Traffic indirectly related data:* We collect the traffic indirectly related data at Shenzhen. It includes weather conditions and PoIs. The weather conditions consist of 16 types, such as sunny, rainy, etc. The PoIs has 659,494 records. Each record includes name, longitude, latitude, primary category, secondary category and address. The primary category has 20 types, such as incorporated business, real estate, financial area, education zone, etc. Each primary category includes different number of secondary category. For example, there are three secondary categories for real estate and twelve secondary categories for education zone. There are 135 secondary categories all together.

2) BASELINES

For evaluation, we use the Root Mean Squared Error (RMSE) and Mean Absolute Errors (MAE). We compare our model with the following baselines:

- *HA:* We predict traffic condition by the average value of history value in the corresponding periods, e.g., 6:00am-6:15am on Monday, its corresponding time spans are all historical time intervals from 6:00am to 6:15am on all historical Monday.
- *ARIMA:* Auto-Regressive Integrated Average is fitted to time series data either to better understand the data or to predict future points in the series [15]
- *SARIMA:* The SARIMA is an extension of ARIMA that explicitly supports univariate time series data with a seasonal component [36].
- *GCN:* We use STGCN [22] as an example. The channels of three layers in STGCN are 64, 32, 128 respectively.

To evaluate each component of our model, we also compare it the difference of fusion layers.

- *GCN-SDF-logits:* GCN-SDF-logits only considers data fusion in spatial domain and the fusion layer is located at the logits layer.
- *GCN-SDF-FM:* GCN-SDF-FM only considers data fusion in spatial domain and the fusion operation is located at the middle layer as depicted in Section III-B.
- *GCN-STDF-logits:* GCN-STDF-logits considers data fusion both in spatial domain and temporal domain. But the fusion layer is located at the logits layer.
- *GCN-STDF-FM:* GCN-STDF-FM considers data fusion both in spatial domain and temporal domain. The fusion layer is located at the middle layer as depicted in Section III-B.

All above methods are evaluated and compared using datasets: Metro and TaxiSZ. All GCN-based networks are trained using fine-tuned hyper-parameters. We use five-fold cross validation for calculating its average performance. All networks have been trained using 50 epochs under the same settings with TensorFlow implementations. We use Adam optimizer [37] to train all networks. For each node, the traffic indirectly related data contains all the features within one-kilometer radius.

TABLE 2. Overall performance.

Model	RMSE		MAE	
	Metro	TaxiSZ	Metro	TaxiSZ
HA	98.65	27.52	50.46	18.37
ARIMA	93.28	23.08	45.25	15.97
SARIMA	90.35	21.48	43.71	14.26
GCN	72.92	13.03	36.19	8.68
GCN-SDF-logits	73.73	13.07	36.47	8.74
GCN-SDF-FM	70.18	12.96	36.12	8.66
GCN-STDF-logits	69.84	13.02	33.86	8.69
GCN-STDF-FM	**68.49**	**12.91**	**33.65**	**8.62**

B. OVERALL COMPARISONS

Table 2 demonstrates the results of STDF and the baselines on the datasets Metro and TaxiSZ. ARIMA gets the worst results because of its low capacity in handling spatio-temporal data prediction. GCN get a better performance than ARIMA. However, GCN-SDF-logits gets a worse results compared with GCN only. Although data fusion is believed to be more effective than the one without data fusion, we can see that data fusion by putting more data into one model may bring negative effects on the model performance. GCN-SDF-FM and GCN-SDF-logits, which only consider the spatial property but ignore the temporal dependency, have much higher RMSE. We call this phenomenon as negative fusion. GCN-SDF-FM and GCN-STDF-FM get a better performance than GCN-SDF-logits and GCN-STDF-logits, which suggests it is better to locate the fusion layer at the middle layer but not at the logits layer. Our proposed model GCN-STDF-FM consistently achieves the best performance on the datasets Metro and TaxiSZ, which shows the effectiveness of using spatial property and temporal property simultaneously. The intuition is that STDF gives the model the ability to capture the dynamic traffic demands and relationships between the node and its surroundings.

C. TRAINING EFFICIENCY AND GENERALIZATION

In order to further investigate the overload caused by adding more data when predicting, we calculate the parameters size and training time consumption (second per epoch) as shown in Table 3. For TaxiSZ dataset, the GCN only model has 1,090,824 parameters and consumes 21.950s seconds per epoch on the training process. Meanwhile, our model only consume 64,640 more parameters, i.e., it cause only 5.59%

TABLE 3. Training efficiency.

Model	Parameter size		Time consumption	
	Metro	TaxiSZ	Metro	TaxiSZ
GCN	289,459	1,090,824	1.848	21.950
GCN-STDF-FM	354,099	1,155,464	1.983	22.879

of parameters increasing. And the training time of our model only cause 0.909 seconds longer than GCN per epoch, which is practical for the real traffic prediction. Similar observations have been also obtained for Metro dataset. GCN has been improved with 6.1% lower testing error by using the method STDF.

D. CASE STUDIES

To understand the performance of STDF, we conduct the following case studies.

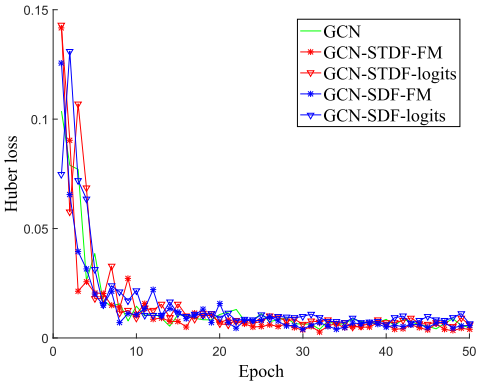


FIGURE 7. Training process.

1) FITTING CAPACITY

Figure 7 demonstrates the comparison of training process of GCN, GCN-STDF-FM, GCN-STDF-logits, GCN-SDF-FM and GCN-SDF-logits. We randomly select one of five-fold cross validation to show the training process. Each network is trained for 50 epoches. The X axis stands for the epoch number. and the Y axis is the loss value. Taking the metro data as an example, we can see that GCN-STDF-FM achieves the lowest training loss and GCN-SDF-logits with the highest training loss. Similar phenomenon can be seen at the testing performance demonstrated at Figure 8 corresponding to Figure 7. It can be clearly observed that STDF provides GCN both (1) enhanced capacity to fit training data as well as (2) the generalizability to adapt testing samples.

2) LEARNING RATE

Configuring the learning rate is challenging and time-consuming. We use five-fold cross validation for searching the best configurations of the learning rate for each experiment by grid search. We set the learning rate to be 0.00001, 0.0001, 0.001, 0.01, 0.1. As observed from Figure 9, the test RMSE reaches to the best performance 68.49 when the learning rate is set to 0.001. The learning rate is fixed to 0.001 in all experiments for STDF.

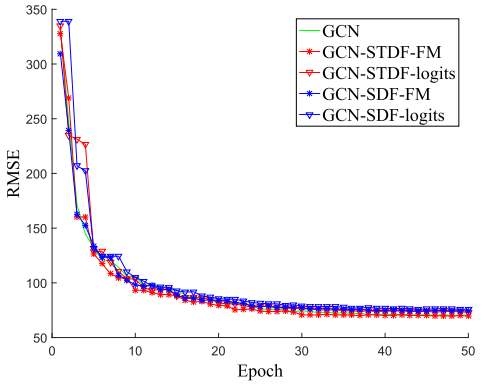


FIGURE 8. Testing performance.

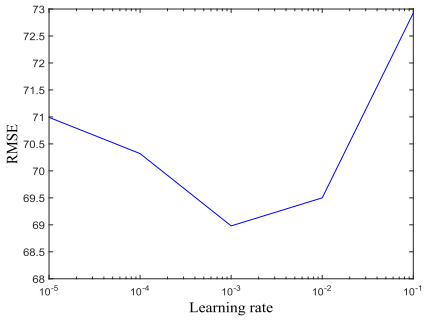


FIGURE 9. Learning rate curve.

VII. CONCLUSION

In this paper, we propose a novel framework STDF for traffic prediction to handle multi-source data fusion. A split-transform-merge strategy is used in STDF. We first separate multi-source data into directly related data and indirectly related data, which are input to GCN and SETON, correspondingly. The feature transformer module is designed to extract spatiotemporal representation for traffic indirectly related data. We get the spatiotemporal representation for traffic directly related data from the middle layer of GCN. This is then followed by a fusion module to combine all features to make final prediction. By using a data sharing strategy, our model is scalable and the overload caused by fusing traffic indirectly related data is acceptable in the real traffic prediction. Experimental results show that our model achieves the best performance compared with other state-of-the-art methods on two real world datasets. In summary, STDF can successfully capture the spatial features changing over time from multi-domain data, which not only can be used into traffic prediction, but also can be applied into other spatiotemporal data prediction. In the future, we will predict the traffic congestion diffusion via representation learning, where the representation vectors are extracted from the fusion layer of STDF. A traffic congestion control policy will be made according to the traffic congestion diffusion model.

REFERENCES

[1] A. M. Nagy and V. Simon, "Survey on traffic prediction in smart cities," *Pervas. Mobile Comput.*, vol. 50, pp. 148–163, Oct. 2018.

[2] L. Zhu, F. R. Yu, Y. Wang, B. Ning, and T. Tang, "Big data analytics in intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 1, pp. 383–398, Jan. 2019.

[3] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1624–1639, Dec. 2011.

[4] B. Zhao, C. Xu, and S. Liu, "A data-driven congestion diffusion model for characterizing traffic in metrocity scales," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2017, pp. 1243–1252.

[5] J. Liu, J. Wan, D. Jia, B. Zeng, D. Li, C.-H. Hsu, and H. Chen, "High-efficiency urban traffic management in context-aware computing and 5G communication," *IEEE Commun. Mag.*, vol. 55, no. 1, pp. 34–40, Jan. 2017.

[6] Y. Zheng, "Methodologies for cross-domain data fusion: An overview," *IEEE Trans. Big Data*, vol. 1, no. 1, pp. 16–34, Mar. 2015.

[7] J. Zhao, Q. Qu, F. Zhang, C. Xu, and S. Liu, "Spatio-temporal analysis of passenger travel patterns in massive smart card data," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 3135–3146, Nov. 2017.

[8] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2016, pp. 785–794.

[9] A. Liaw and M. Wiener, "Classification and regression by randomforest," *R News*, vol. 2, no. 3, pp. 18–22, 2002.

[10] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3146–3154.

[11] T. G. Dietterich, "Ensemble learning," *The Handbook of Brain Theory Neural and Network*, vol. 2. 2002, pp. 110–125.

[12] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017.

[13] 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 *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2019.

[14] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018.

[15] B.-S. Chen, S.-C. Peng, and K.-C. Wang, "Traffic modeling, prediction, and congestion control for high-speed networks: A fuzzy AR approach," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 5, pp. 491–508, 2000.

[16] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results," *J. Transp. Eng.*, vol. 129, no. 6, pp. 664–672, Nov. 2003.

[17] W. Min and L. Wynter, "Real-time road traffic prediction with spatio-temporal correlations," *Transp. Res. C, Emerg. Technol.*, vol. 19, no. 4, pp. 606–616, Aug. 2011.

[18] X. Gao, Y. Zhao, Ł. Dudziak, R. Mullins, and C.-z. Xu, "Dynamic channel pruning: Feature boosting and suppression," 2018, *arXiv:1810.05331*. [Online]. Available: http://arxiv.org/abs/1810.05331

[19] M. Aqib, R. Mehmood, A. Alzahrani, I. Katib, A. Albeshri, and S. M. Altowaijri, "Smarter traffic prediction using big data, in-memory computing, deep learning and GPUs," *Sensors*, vol. 19, no. 9, p. 2206, 2019.

[20] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transp. Res. C, Emerg. Technol.*, vol. 54, pp. 187–197, May 2015.

[21] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, 2014.

[22] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 3634–3640, doi: 10.24963/ijcai.2018/505.

[23] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 922–929.

[24] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," 2017, *arXiv:1707.01926*. [Online]. Available: http://arxiv.org/abs/1707.01926

[25] J. Bleiholder and F. Naumann, "Data fusion," *ACM Comput. Surveys*, vol. 41, no. 1, p. 1, 2009.

[26] Z. Wang, D. Zhang, X. Zhou, D. Yang, Z. Yu, and Z. Yu, "Discovering and profiling overlapping communities in location-based social networks," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 44, no. 4, pp. 499–509, Apr. 2014.

[27] D. Yang, D. Zhang, Z. Yu, and Z. Yu, "Fine-grained preference-aware location search leveraging crowdsourced digital footprints from LBSNs," in *Proc. ACM Int. joint Conf. Pervas. ubiquitous Comput. (UbiComp)*, 2013, pp. 479–488.

[28] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 4, pp. 217–228, Oct. 2005.

[29] A. Samant and H. Adeli, "Feature extraction for traffic incident detection using wavelet transform and linear discriminant analysis," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 15, no. 4, pp. 241–250, Jul. 2000.

[30] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Müller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," 2016, *arXiv:1604.07316*. [Online]. Available: http://arxiv.org/abs/1604.07316

[31] D. Datcu and L. J. Rothkrantz, "Semantic audio-visual data fusion for automatic emotion recognition," in *Emotion Recognition: A Pattern Analysis Approach*. 2014, pp. 411–435.

[32] E. Bozkurt, E. Erzin, C. C. E. Erdem, and A. T. Erdem, "Improving automatic emotion recognition from speech signals," in *Proc. 10th Annu. Conf. Int. Speech Commun. Assoc.*, 2009.

[33] C. M. Lee, S. S. Narayanan, and R. Pieraccini, "Combining acoustic and language information for emotion recognition," in *Proc. 7th Int. Conf. Spoken Lang. Process.*, 2002.

[34] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "DeepFM: A factorization-machine based neural network for CTR prediction," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 1–8, doi: 10.24963/ijcai.2017/239.

[35] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Appl. Comput. Harmon. Anal.*, vol. 30, no. 2, pp. 129–150, Mar. 2011.

[36] T.-M. Choi, Y. Yu, and K.-F. Au, "A hybrid SARIMA wavelet transform method for sales forecasting," *Decis. Support Syst.*, vol. 51, no. 1, pp. 130–140, Apr. 2011.

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980