Dynamic Spatio-temporal Graph-based CNNs for Traffic Flow Prediction

KEN CHEN [1], FEI CHEN [1], BAISHENG LAI[2], ZHONGMING JIN[2], YONG LIU[3], KAI LI [1], LONG WEI [2], PENGFEI WANG[2], YANDONG TANG [1], JIANQIANG HUANG[2], XIAN-SHENG HUA[2](Fellow, IEEE)

[1]Sichuan Highway Transportation & Communication Project Co., Ltd., Chengdu, China
[2]Alibaba Damo Academy, Alibaba Group, Hangzhou, China
[3]Alibaba Cloud, Alibaba Group, Hangzhou, China

ABSTRACT Forecasting future traffic flows from previous ones is a challenging problem because of their complex and dynamic nature of spatio-temporal structures. Most existing graph-based CNNs attempt to capture the static relations while largely neglecting the dynamics underlying sequential data. In this paper, we present dynamic spatio-temporal graph-based CNNs (DST-GCNNs) by learning expressive features to represent spatio-temporal structures and predict future traffic flows from surveillance video data. In particular, DST-GCNN is a two stream network. In the flow prediction stream, we present a novel graph-based spatio-temporal convolutional layer to extract features from a graph representation of traffic flows. Then several such layers are stacked together to predict future flows over time. Meanwhile, the relations between traffic flows in the graph are often time variant as the traffic condition changes over time. To capture the graph dynamics, we use the graph prediction stream to predict the dynamic graph structures, and the predicted structures are fed into the flow prediction stream. Experiments on real datasets demonstrate that the proposed model achieves competitive performances compared with the other state-of-the-art methods.

INDEX TERMS Graph Neural Networks, Traffic Forecasting, Time Series Regression

I. INTRODUCTION

THE goal of traffic flow forecasting is to predict the future traffic flows based on previous flows measured by sensors, which is one of the most challenging problems in Intelligent Transportation System (ITS). In the context of traffic flow forecasting, "traffic flows" or "traffic volumes" mean the number of cars recorded by a sensor network in a period of time. Accurate traffic forecasting can enable individuals and policy makers to make decisions on route planning and traffic control.

Advanced algorithms that can model the interactions between dynamic traffic flows are required to predict their future trends. In literature, data-driven approaches have attracted many research attentions. For example, statistical methods such as autoregressive integrated moving average (ARIMA) [1] and its variants [2] are well studied. The performance of such methods are limited because their capacity is insufficient to model the complex nonlinear dependency among traffic flows in either spatial or temporal contexts. Recently, deep learning methods have shown promising results

in dynamic prediction over sequential data, including stacked autoencoder (SAE) [3], DBN [4], LSTM [5] and CNN [6]. Although these methods made some progress in modeling complex patterns in sequential data, they have not yet fully explored both spatial and temporal structures of traffic flows in an integrated fashion.

Several methods [7], [8] attempt to model the traffic flows by unrolling static graphs through time where each vertex denotes the reading of flows at a given location and edges represent how the flows at two locations would affect each other. These works show that the graph structure is capable of describing the spatio-temporal dependency between flows. However, they usually have to assume that the graph structures, especially the relations between flows at different locations, do not change over time. It implies traffic conditions are time-invariant , which is not true in the real world.

To address this problem, we propose a dynamic spatio-temporal graph based CNN (DST-GCNN), which can model both the dynamics of traffic flows and their correlations. The contributions of this paper are threefold.

- We propose a novel spatio-temporal graph-based convolutional layer that is able to jointly extract both spatio and temporal information from the traffic flow data. This layer consists of two factorized convolutions applied to spatial and temporal dimensions respectively, which significantly reduces computations and can be implemented in a parallel way. Then, we build a hierarchy of stacked graph-based convolutional layers to extract expressive features and make traffic flow predictions.

- We will also learn the evolving graph structures that can adapt to the fast-changing traffic conditions over time. The learned graph structures can be seamlessly integrated with the stacked graph-based convolutional layers to make accurate traffic flow predictions.

- We evaluate the proposed model on both traffic video dataset and the public Beijing taxi dataset. Experimental results demonstrate that DST-GCNN outperforms the state-of-the-art methods.

II. RELATED WORK

The study of traffic flow forecasting can trace back to 1970s [9]. From then on, a large number of methods have been proposed, and a recent survey paper comprehensively summarizes the methods [10]. Early methods were often based on simulations, which were computationally demanding and required careful tuning of model parameters. With modern real-time traffic data collection systems, data-driven approaches have attracted more research attentions. In statistics, a family of autoregressive integrated moving average (ARIMA) models [1] are proposed to predict traffic flows. However, these autoregressive models rely on the stationary assumption on sequential data, which fails to hold in real traffic conditions that vary over time. In [11], Intrinsic Gaussian Markov Random Fields (IGMRF) are developed to model both the season flows and the trend flows, which is shown to be robust against noise and missing data. Some conventional learning methods including Linear SVR [12] and random forest regression [13] have also been tailored to solve traffic flow prediction problem. Nevertheless, these shallow models depend on hand-crafted features and can not fully explore complex spatio-temporal patterns among the big traffic data, which greatly limits their performances.

With the development of deep learning, various network architectures have been proposed for predicting traffic flows. Early attempts include SAE [3] and DBN [4], but neither are effective in modeling the spatio-temporal dependency between traffic flows. To capture the short and long temporal dependency, LSTMs are used to model the evolution of traffic flows [5]. However, the typical LSTM model is unable to model the spatial correlations which play an important role in making a spatially coherent prediction on the traffic flows. To close this gap, hybrid models where temporal models such as LSTM and GRU are combined with spatial models like 1D-CNN [14] and graphs [7] are proposed and achieve impressive performances. Nevertheless, recurrent models are restricted to process sequential data successively one-after-one, which limits the parallelization of underlying computations. In contrast, the proposed model utilizes convolutions to capture both spatial and temporal data dependencies, which can reach much more efficiency than the compared recurrent models.

Two recent state-of-the-art methods are DCRNN [15] and STGCN [8]. They show appealing results on public datasets. But DCRNN is less efficient as it involves recurrent feedforward. STCGN use fixed affinity matrix that is not suitable for dynamic traffic environments. In contrast, our paper provides an efficient and dynamic method for better traffic prediction.

III. PRELIMINARIES

A. STRUCTURED INFORMATION EXTRACTION

To collect traffic flow data, multiple sensors including loop detectors [16], radar [8] and GPS trajectories [11] can be leveraged. However, they are either incomplete or unable to capture fine-grained trajectory flows of individual vehicles. In contrast, we propose in this paper to use surveillance videos to record and predict traffic flows. Thanks to the fast development of computer vision technologies, we can not only track individual flows of vehicles but also link their identities across different cameras over time.

In specific, we first use SSD [17] and KCF [18] to identify and track vehicle instances in videos. Then the plate numbers are recognized so that the same vehicles can be targeted across cameras. For each instance, we record the detected time, the camera location and its plate number, which are referred to as structured information in this paper. With them, at each location, we can count traffic volumes in a period of time. Meanwhile, by tracking plate numbers, we can estimate the average time to travel from one location to another. The computed traffic volumes and travel time are the inputs of our method.

It is worth noting that estimating the travel time across different locations does not require to recognize the plate numbers for all vehicles. The estimated travel time only need to be computed over those with recognized plate numbers, which greatly simplifies the problem.

B. MATHEMATICAL NOTATIONS

Suppose at each time t, we have volumes of traffic flows $\mathbf{F}_t \in \mathbb{R}^{C_0 \times N}$ and travel time $\mathbf{T}_t \in \mathbb{R}^{N \times N}$ data, where N is number of locations and C_0 is the number of input channels. The channels represent different directions of traffic volumes at a location. Our method uses T_P previous traffic volumes $\{\mathbf{F}_{t-T_P+1}, ..., \mathbf{F}_t\}$ to forecast future volumes $\mathbf{F}_{t+T_F}$ after T_F time steps. For simplicity, we use a tensor $\mathcal{X}_t \in \mathbb{R}^{C_0 \times T_P \times N}$ to denote $\{\mathbf{F}_{t-T_P+1}, ..., \mathbf{F}_t\}$. Without ambiguity, the subscript t may be ignored in the rest of paper.

To model the complex dependency among traffic volumes, in DST-GCNN, we use an undirected graph $\mathbf{G} = (\mathbf{V}, \mathbf{A})$ to represent the traffic volumes, where the vertex set $\mathbf{V}$ represents traffic volumes at different locations and the affinity matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ depicts the connectivity between vertices. We derive the affinity matrix from the travel time

such that $\mathbf{A}_{ij} = exp(-\mathbf{T}_{ij}/\sigma)$, where $\mathbf{T}_{ij}$ is the travel time between location i and j. Therefore, the historical traffic volumes $\mathcal{X}_t$ can be represented as stacked graph frames.

IV. METHOD

The proposed DST-GCNN framework can model the complex spatio-temporal dependency between traffic flows and the fast-evolving traffic conditions. It takes three inputs: the previous traffic volumes represented as stacked graph frames, the previous traffic conditions represented as a series of affinity matrices and auxiliary information. Then these types of information are fed to a two-stream network. The graph prediction stream predicts the traffic conditions while the flow prediction stream forecasts evolutions of traffic flows given the predicted traffic conditions. The overall architecture of DST-GCNN is presented in Figure 1. In the following subsections, we describe the two streams in details.

A. FLOW PREDICTION STREAM

In this subsection, we introduce the structure of the flow prediction stream that is the main sub-network to perform prediction. First, we present the building block of this stream, which is a novel Spatio-temporal Graph-based Convolutional Layer (STC) that works with spatio-temporal graph data. Then we build a two-step hierarchical model using STC layers to predict traffic flows.

1) Spatio-temporal Graph-based Convolution

The CNN is a popular tool in computer vision as it is powerful to extract hierarchy features expressive in many high-level recognition and prediction tasks. However, it cannot be directly applied to process the structured graph data like in our task. Therefore, we propose a novel layer that works with spatio-temporal graph data and is also as efficient as conventional convolutions.

Inspired by [19] that factorizes convolutions along two separate dimensions, we also present two factorized convolutions applied to spatial and temporal dimensions respectively, in a hope to reduce computational overhead. They form the proposed Spatio-temporal Graph-based Convolutional Layer (STC), whose structure is shown in Figure 2. The input to a STC layer contains a sequence of graph structured feature maps organized by their timestamps and channels. Each graph is first convolved spatially to extract its spatial feature representation, and then features of multiple graphs are fused by a temporal convolution in a sliding time-window. In this way, both spatial and temporal information are merged to yield a dynamic feature representation for predicting future flows.

Spatial Convolution

Let us define the spatial convolution on a given graph $\mathbf{G} = (\mathbf{V}, \mathbf{A})$ first. The diagonal degree matrix and the graph Laplacian are defined as $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$ and $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ respectively. Then the Singular Value Decomposition (SVD) is applied to Laplacian as $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$,

where $\mathbf{U}$ consists of eigen vectors and $\mathbf{\Lambda}$ is a diagonal matrix of eigen values. The matrix U is the Graph Fourier Transform matrix, which transforms an input graph signal $\mathbf{x} \in \mathbb{R}^N$ to its frequency domain $\mathbf{U}\mathbf{x} \in \mathbb{R}^N$. With the same notation in [20], the convolution of a graph signal $\mathbf{x}$ with filter $\mathbf{g} \in \mathbb{R}^N$ on $\mathbf{G}$ is defined as

$$\mathbf{x} *_{\mathbf{G}} \mathbf{g} = \mathbf{U}^T(\mathbf{U}\mathbf{g} \odot \mathbf{U}\mathbf{x}), \qquad (1)$$

where $\odot$ is the element-wise product.

Let's define $\mathbf{w} = \mathbf{U}\mathbf{g}$ as the filter in frequency domain, then the convolution can be rewritten as

$$\mathbf{x} *_{\mathbf{G}} \mathbf{g} = \mathbf{U}^T(diag(\mathbf{w})\mathbf{U}\mathbf{x}), \qquad (2)$$

The above graph convolution requires filter $\mathbf{w}$ to have the same size as input signal $\mathbf{x}$, which would be inefficient and hard to train when the graph has a large size. To make the filter "localized" as in CNN, $diag(\mathbf{w})$ can be approximated as polynomials of $\mathbf{\Lambda}$ [21] so that $diag(\mathbf{w}) = \sum_{k=0}^{K-1} \theta_k \mathbf{\Lambda}^k$ and Eq 2 can be rewritten as

$$\mathbf{x} *_{\mathbf{G}} \mathbf{g} = \sum_{k=0}^{K-1} \theta_k \mathbf{L}^k \mathbf{x}. \qquad (3)$$

Now the trainable parameters become $\theta \in \mathbb{R}^K$ whose size is restricted to K. In addition, a node is only supported by its $(K-1)$ neighbors [22].

Then we use the convolution operation above to define the spatial convolution in STC layer. When computing the spatial convolution between feature map $\mathcal{X}^l \in \mathbb{R}^{C_l \times T_P \times N}$ and kernel $\mathcal{W}^l \in \mathbb{R}^{C_l \times T_P \times K}$ in the l-th layer of DST-GCNN, where C_l is the channel number, the graph-based convolution defined above is applied to individual graph frame separately. In specific, each graph feature $\mathcal{X}^l_{c,p} \in \mathbb{R}^N$ at c-th channel and p-th time step is individually filtered such that

$$\mathcal{Z}^l_{c,p} = \mathcal{X}^l_{c,p} *_{\mathbf{G}} \mathcal{W}^l_{c,p}, \qquad (4)$$

where $\mathcal{W}^l_{c,p} \in \mathbb{R}^K$ and $\mathcal{Z}^l_{c,p} \in \mathbb{R}^N$ are the individual kernel and filtered output at the c-th channel and p-th time step, while tensor $\mathcal{Z}^l \in \mathbb{R}^{C_l \times T_P \times N}$ is the whole output.

Temporal Convolution

At each time, after the spatial convolution, traffic flows are fused on the underlying graph, resulting in a multi-layered feature tensor $\mathcal{Z}^l$ compactly representing individual traffic flows and their spatial interactions.

However, information across time steps is still isolated. To obtain spatio-temporal features, many previous methods [5], [23], [24] are based on recurrent models, which process sequential data iteratively step-by-step. Consequently, the information of current step is processed only when the information of all previous steps are done, which limits the efficiency of recurrent models.

To make temporal operations as efficient as a convolution, we perform a conventional convolution along the time dimension to extract the temporal relations, named after temporal convolution. For a feature tensor $\mathcal{Z}^l$ of size $[C_l, T_P, N]$,
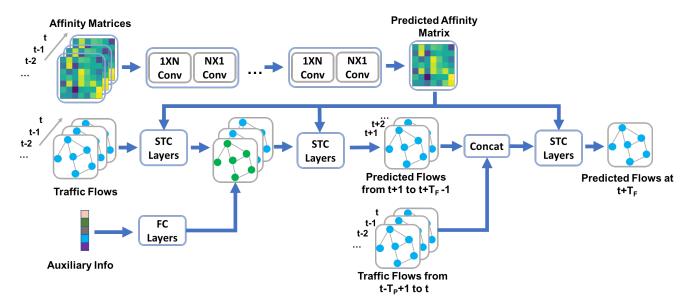
FIGURE 1: The overview of the proposed framework. The network consists of two streams, the first stream predicts the dynamic traffic conditions which are encoded in an affinity matrix. The second stream, equipped with the predicted traffic conditions and the proposed STC layers, first predicts future flows from $t + 1$ to $t + T_F - 1$, then predicts the target future flows at $t + T_F$.

its convolution with kernel $\mathcal{K}^l$ of size $[C_l, C_{l+1}, Q, 1]$ is performed,

$$\mathcal{X}^{l+1} = \mathcal{Z}^l * \mathcal{K}^l, \tag{5}$$

where Q is the size of time window. To keep the size of the time dimension unchanged, we pad $(Q-1)/2$ zeros on both sides of the time dimension.

Putting Together
By combining Eq. 4 and Eq. 5, we have the following definition of spatio-temporal graph-based convolution:

$$\mathcal{X}^{l+1} = STC(\mathcal{X}^l, \mathcal{W}^l, \mathcal{K}^l, \mathbf{G}), \tag{6}$$

whose structure is shown in Figure 2.

We now analyse the efficiency of our factorized convolution. Without such factorization, one needs to build a graph with $N \times C_l \times T_P$ nodes to capture both spatial and temporal structures, making the graph convolution in Eq. 2 have complexity of $\mathcal{O}(N^2 C_l^2 T_P^2)$. While our STC layer builds $C_l \times T_P$ graphs with N nodes and separates spatial and temporal convolutions, has complexity of $\mathcal{O}(N^2 C_l T_P + N C_l C_{l+1} T_P Q)$, which is much more efficient.

2) Two-step Prediction
The STC layers are able to jointly extract both spatial and temporal information from the sequence of traffic flows. We can build a hierarchical model using such layers to extract features and predict future flows from previous flows $\{\mathbf{F}_{t-T_P+1}, ..., \mathbf{F}_t\}$. A straight way is to directly predict future traffic volume $\mathbf{F}_{T_F}$ after T_F intervals as existing methods [5], [8], [21]. This one-step prediction scheme is simple but has two disadvantages. First, it only uses ground truth data at $t + T_F$ to train the model but neglects those between $t + 1$ and $t + T_F - 1$. Second, when T_F is large, it

is hard for one-step methods to capture traffic trends for such a long time, since the input and the future volumes may be very different.

To solve the above issues, we propose a new prediction scheme that divides the prediction problem into two steps. In the first step, we use previous flows $\{\mathbf{F}_{t-T_P+1}, ..., \mathbf{F}_t\}$ to predict future volumes between $t + 1$ and $t + T_f - 1$, which are called "close future flows". During the training phase, the predicted "close future flows" are supervised by ground truth at the corresponding time period. As a result, ground truth data between $t + 1$ and $t + T_F - 1$ is imposed into training procedure. In the second step, the "target future flows" at time $t + T_F$ is predicted by considering both previous flows and the predicted "close future flows". Compared with one-step methods, the prediction of "target future flows" is easier now since it utilizes "close future flows" and it only predicts one step further. The two-step prediction scheme is shown in the second path in Figure 1.

Let's denote the models of the first step and the second step as $\mathcal{M}_{S1}$ and $\mathcal{M}_{S2}$ respectively. These two model both stacks several STC layers for prediction. The loss function of two-step prediction can be written as:

$$L_{two-step} = \|\mathcal{Y}_t - \hat{\mathcal{Y}}_t\|^2 + \|\mathbf{F}_{t+T_F} - \mathcal{M}_{S2}(\mathcal{X}_t, \hat{\mathcal{Y}}_t, \mathbf{\Theta}_{S2})\|^2, \tag{7}$$

where $\hat{\mathcal{Y}}_t = \mathcal{M}_{S1}(\mathcal{X}_t, \mathbf{\Theta}_{S1})$ is the predicted "close future flows" and $\mathcal{Y}_t = \{\mathbf{F}_{t+1}, ..., \mathbf{F}_{t+T_F-1}\}$ is the ground truth. $\mathbf{\Theta}_{S1}$ and $\mathbf{\Theta}_{S2}$ are parameters of two models respectively.

3) Auxiliary Information Embedding
Except for previous flows, some auxiliary information like time, the day of week and weather are useful to predict future flows. The influence of such information is studied in

Graph Structured Feature Maps   **Spatial Convolution**   **Temporal Convolution**
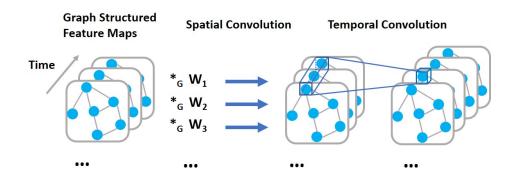
FIGURE 2: The structure of spatio-temporal graph-based convolution. It consists of two convolutions that are applied on spatial and temporal dimensions respectively.

[6], [11]. For example, weekdays and weekends have very different transit patterns and a thunder storm can suddenly reduce the traffic volumes.

To make full use of such auxiliary information, we embed them into the traffic flow prediction network. We first encode these information into one-hot vectors. Then these one-hot vectors are concatenated and we use several fully connected layers to extract a feature vector. The feature vector is later reshaped so that it can be concatenated with traffic flow feature maps. Finally, the concatenated features are fed into prediction modules, as shown in Figure 1.

B. GRAPH PREDICTION STREAM

In this subsection, we introduce the other stream in the framework, which is named as the graph prediction stream. Previous methods [8], [20], [23] that model spatio-temporal graphs assume that the graph structure of spatio-temporal data is fixed without temporal evolutions. However, in real world applications, the graph structures are dynamic. For instance, in the traffic prediction problem, traffic conditions are time-variant, implying that the connectivities between vertices in graphs change over time. In order to model such dynamics, we introduce a stream in the framework to predict such time-variant graph structures.

In particular, at each time t, we have a graph structure $\mathbf{G}_t$ for STC layers in the model as a function of time t. It reflects the average traffic condition in the period between time $t - T_P + 1$ and $t + T_F$. $\mathbf{G}_t$ can not be directly computed since the future travel time during $t+1$ to $t+T_F$ is unavailable in the test phase. To address this problem, we introduce another path in the network to predict graph structure $\mathbf{G}_t$ from previous travel time data $\{\mathbf{T}_{t-T_P+1}, ..., \mathbf{T}_t\}$. Specifically, $\{\mathbf{T}_{t-T_P+1}, ..., \mathbf{T}_t\}$ are first converted to affinity matrices to construct a tensor $\mathcal{S}_t = \{\mathbf{A}_{t-T_P+1}, ..., \mathbf{A}_t\} \in \mathbb{R}^{T_P \times N \times N}$, then it is fed into a sub-network $\mathcal{M}_G$ to predict a new affinity matrix $\hat{\mathbf{A}}_t = \mathcal{M}_G(\mathcal{S}_t, \boldsymbol{\Theta}_G)$ representing for the average traffic condition during $t - T_P + 1$ and $t + T_F$, where $\boldsymbol{\Theta}_G$ is parameter of $\mathcal{M}_G$.

During training, the graph prediction stream is supervised

by minimizing the following loss function

$$L_{dynamic} = \sum_t \|\hat{\mathbf{A}}_t - \bar{\mathbf{A}}_t\|_1, \tag{8}$$

where $\bar{\mathbf{A}}_t \in \mathbb{R}^{N \times N}$ is the ground truth average affinity matrix during $t - T_P + 1$ and $t + T_F$. L_1 norm is used to avoid the loss from being dominated by some large errors. The Laplacian of $\hat{\mathbf{A}}_t$ is then computed and fed into STC layers. In this way, the prediction model takes the dynamic traffic conditions into consideration, thus it is able to make more accurate predictions on future traffic flows.

To model the relations of previous affinity matrices, a model with global field of view is required since entries of affinity matrices have "global" correlations. For instances, $\mathbf{A}_{ij}$ and $\mathbf{A}_{ji}$ is closely related no matter how apart they are located in $\mathbf{A}$. Thus, we stack multiple pairs of convolutional layers, where each pair consists of convolutional layers of kernel sizes $[1, N]$ and $[N, 1]$ respectively to get the large spatial extent, as shown in the first stream of Figure 1.

C. THE WHOLE MODEL

By combining the two paths, we get the full model of DST-GCNN shown in Figure 1. The loss function of the complete model is

$$L = L_{two-step} + L_{dynamic} \tag{9}$$

It is worth noting that DST-GCNN is a general method to extract features on spatio-temporal graph structured data, it can be applied to not only traffic flow prediction tasks, but also other more general regression or classification tasks on graph data, especially when the graph structure is dynamic. For instance, it can be adapted to skeleton-based action recognition or pose forecasting tasks with minor modification.

V. EXPERIMENTS

In this section, we present a series of experiments to assess the performance of the proposed methods. We first introduce the datasets and the implementation details of DST-GCNN. Then we conduct ablation experiments to evaluate the effectiveness of components in DST-GCNN. At last, our method are compared with state-of-the-art methods on these datasets.

A. DATASET AND EVALUATION METRICS

Our experiments are conducted on two public datasets: METR-LA [16] and TaxiBJ [11], and our collected dataset CD-HW.

We first introduce two public available **METR-LA** [16] and **TaxiBJ** [11]. METR-LA is a large-scale dataset collected from 1500 traffic loop detectors in Los Angeles country road network. This dataset includes speed, volume and occupancy data, covering approximately 3,420 miles. As [15], we choose four months of traffic speed data from Mar 1st 2012 to Jun 30th 2012 recorded by 207 sensors for our experiment. The traffic data are aggregated every 5 minutes with one direction. TaxiBJ Dataset are obtained from taxis' GPS trajectories in Beijing during 1st March to 30th June 2015. The authors partition Beijing into 26 high-level regions and traffic volumes are aggregated in every 30 minutes, with two directions {In, Out}. Besides crowd flows, it also includes weather conditions that are categorized into good weather (sunny, cloudy) and bad weather (rainy, storm, dusty).

We also collect a new dataset that contains speed data recorded along highway around Chengdu city, China. This dataset is named as Chengdu Highway (**CD-HW**) Dataset. CD-HW dataset is capured by 1,692 roadside sensors during 1 Nov to 30 Nov 2019. Their locations are shown in Figure 3. Data before 25th Nov are used for training and the remaining for test. The speed data are colleced every 10 minutes with one direction.

FIGURE 3: Distribution of sensors on highway around Chengdu City in the CD-HW dataset.

For evaluation, we use the Root Mean Squared Error (RMSE) metric, the Mean Absolute Percentage Error (MAPE) and the Mean Absolute Error (MAE), which are

defined as below:

$$
\begin{aligned}
RMSE &= \frac{1}{N_T} \sum_{t=1}^{N_T} \sqrt{\frac{1}{N} \sum_{i=1}^{N} (\hat{\mathbf{F}}_{t,i} - \mathbf{F}_{t,i})^2}, \\
MAPE &= \frac{1}{N_T} \sum_{t=1}^{N_T} \frac{1}{N} \sum_{i=1}^{N} |\frac{\hat{\mathbf{F}}_{t,i} - \mathbf{F}_{t,i}}{\mathbf{F}_{t,i}}|, \\
MAE &= \frac{1}{N_T} \sum_{t=1}^{N_T} \frac{1}{N} \sum_{i=1}^{N} |\hat{\mathbf{F}}_{t,i} - \mathbf{F}_{t,i}|,
\end{aligned} \quad (10)
$$

where $\hat{\mathbf{F}}_{t,i}$ and $\mathbf{F}_{t,i}$ are the predicted and ground truth traffic volumes (speed) at time t and location i.

B. IMPLEMENTATION DETAILS

Models $\mathcal{M}_{S1}$ and $\mathcal{M}_{S2}$ presented in subsection IV-A2 consist of three STC layers with 8, 16, 32 channels respectively. A ReLU layer is inserted between two STC layers to introduce nonlinearity as CNNs. Another ReLU layer is added after the last STC layer to ensure non-negative prediction. In spatial convolution of STC layer, the order K of polynomial approximation is set to be 5 and the temporal convolution kernel size is set to be 5×1. The graph prediction stream $\mathcal{M}_G$ consists of three pairs of $1 \times N$ and $N \times 1$ convolutional layers with 16 channels. The auxiliary information is encoded by two fully connected layers with 32 and $N \times N \times C \times T_p$ output neurons respectively, so that the output can be reshaped and concatenated with flow features. In the training procedure, we first pre-train the dynamic graph learning sub-network for 10 epochs and jointly train the whole model for 100 epochs. The model is trained by SGD with momentum. The first 50 epochs take a learning rate of 10^{-2} and the last 50 epochs use 10^{-3}. Finally, the framework is implemented by PyTorch.

C. ABLATION STUDY

To investigate the effectiveness of each component, we first build a plain **baseline model** which stacks three STC layers as $\mathcal{M}_{S1}$ while uses one-step prediction scheme, keeps graph structure fixed and does not use auxiliary information. The static graph structure is calculated by averaging all traffic time in training set. Then different configurations are tested, including:

- The baseline model denoted as denoted as **Basel**;
- The baseline model with auxiliary information embedding (AE), denoted as **Basel+AE**;
- The above configuration plus dynamic graph learning (DG), denoted as **Basel+AE+DG**;
- The above configuration plus two-step prediction (TP) introduced in IV-A2, which is the full model denoted as **Basel+AE+DGL+TP** or **DST-SCNN**.

The experimental results evaluated on the TaxiBJ test set of all configurations are reported in Table 1. We predict two time steps ahead in all configurations. We can observe that each proposed component consistently reduces the prediction errors and the full model achieves the best performance. The

Method	Out Volumes			In Volumes		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE
Basel	10.49	13.48	13.11%	10.71	14.44	14.46%
Basel+AE	10.24	13.18	12.81%	10.41	13.96	14.35%
Basel+AE+GP	10.03	12.88	12.75%	10.40	13.94	14.39%
DST-GCNN	**9.93**	**12.78**	**12.56%**	**10.24**	**13.78**	**14.02%**

TABLE 1: Performance comparison of our models with different configurations on TaxiBJ dataset.

TABLE 2: Performance comparison of different methods on the METR-LA dataset. MAE, RMSE and MAPE(%) metrics are compared for different predicting horizons. Bold numbers indicate the best results.

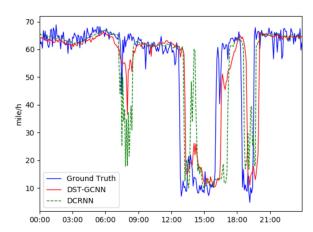| T	Metric	ARIMA	FNN	FC-LSTM	DCRNN	STGCN	DST-GCNN
	MAE	3.99	3.99	3.44	2.77	2.87	**2.68**
15 min	RMSE	8.21	7.94	6.30	5.38	5.54	**5.35**
	MAPE	9.6	9.9	9.6	7.3	7.4	**7.2**
	MAE	5.15	4.23	3.77	3.15	3.48	**3.01**
30 min	RMSE	10.45	8.17	7.23	6.45	6.84	**6.23**
	MAPE	12.7	12.9	10.9	8.8	9.4	**8.5**
	MAE	6.90	4.49	4.37	3.60	4.45	**3.41**
60 min	RMSE	13.23	8.69	8.69	7.59	8.41	**7.47**
	MAPE	17.4	14.0	13.2	10.5	11.8	**10.3**

FIGURE 4: Traffic speed prediction during a day on METR-LA dataset.

results demonstrate that the auxiliary information embedding, the graph prediction stream and the two-step prediction scheme are all beneficial and complementary to each other. The combination of them accumulates the advantages, therefore achieves the best performance.

D. EXPERIMENTS ON METR-LA DATASET

In this subsection, we evaluate the prediction performance of DST-GCNN and the compared methods on METR-LA dataset. We compare DST-GCNN with five different methods, including: 1) Auto-Regressive Integrated Moving Average (ARIMA), which is a well-known method for time-series data forecasting and is widely used in traffic prediction; 2) Feed Forward Neural Network (FNN) with two hidden layers and L2 regularization. 3) Recurrent Neural Network with fully connected LSTM hidden units (FC-LSTM) [24]; 4) DCRNN [15]. This is a recent method which utilizes diffusion convolution and achieves decent results on METR-LA; 5) STGCN [8]. This is a spatio-temporal graph convolutional

networks that uses a fixed affinity matrix.

Table 2 shows the comparison results on METR-LA dataset. For all predicting horizons and all metrics, our method outperforms both traditional statistical approaches and deep learning based approaches. This demonstrates the consistency of our method's performance for both short-term and long-term prediction.

In Figure 4, we also show the qualitative comparison of prediction in a day on the METR-LA dataset. It shows that DST-GCNN can capture the trends of morning peak and evening rush hour better. As can be seen, DST-GCNN predicts the start and end of peak hours which are closer to the ground truth. In contrast, DCRNN does not catch up with the change of traffic data.

E. EXPERIMENTS ON TAXIBJ DATASET

We also compare the proposed methods with the state-of-the-arts on TaxiBJ dataset. The compared methods include: 1) Seasonal ARIMA (SARIMA); 2) vector auto-regression model (VAR); 3) FCCF [25]; 4) FC-LSTM [24]; and 5) DCRNN [15]. FCCF utilizes both volume data and auxiliary information including time and weather. Note that we follow the experiments in FCCF that only predict volumes in the next step (30 min later), thus the two-step prediction in our model is not applied. The results by FCCF, SARIMA and VAR were reported in [25]. Since only the RMSE results are provided for SARIMA, VAR and FCCF, we compare with these three methods in terms of RMSE metric. For FC-LSTM and DCRNN, we use their default experimental settings in the corresponding papers, and the results are compared in terms of RMSE, MAP and MAPE. We show the results in Table 3.

From Table 3, we can see that the proposed DST-GCNN achieves the best performance. The comparison results suggests that the proposed STC layer combined with the graph prediction stream is very effective at future traffic prediction. Although the two-step prediction strategy is not utilized in

Method	Out Volumes			In Volumes		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE
SARIMA	-	21.2	-	-	18.9	-
VAR	-	15.8	-	-	15.8	-
FCCF	-	14.2	-	-	14.1	-
FC-LSTM	11.32	14.4	13.67%	11.92	15.3	17.30%
DCRNN	10.49	13.8	13.11%	10.71	14.5	14.46%
DST-GCNN	**9.38**	**12.0**	**11.9%**	**9.3**	**12.62**	**13.27%**

TABLE 3: Performance comparison of different methods on the TaxiBJ dataset. MAE, RMSE and MAPE(%) metrics are compared for different predicting horizons. Bold numbers indicate the best results.

the case of predicting one-step ahead, our method still models the spatio-temporal dependency and the dynamic graph structure robustly.

F. EXPERIMENTS ON CD-HW DATASET

In this subsection, we evaluate the prediction performance of DST-GCNN and the compared methods on CD-HW dataset. Because deep learning methods have shown better performance than traditional methods, we only compare our methods with DCRNN and STGCN. Table 4 shows the comparison results.

TABLE 4: Performance comparison of different methods on the CD-HW dataset. MAE, RMSE and MAPE(%) metrics are compared for different predicting horizons. Bold numbers indicate the best results.

T	Metric	DCRNN	STGCN	DST-GCNN
	MAE	7.92	7.76	**6.33**
30 min	RMSE	12.10	11.77	**10.18**
	MAPE	12.4	12.2	**10.9**
	MAE	9.22	8.82	**7.91**
60 min	RMSE	14.32	13.61	**12.32**
	MAPE	16.3	15.9	**14.3**

We can observe that our method outperforms the recently deep learning based approaches DCRNN and STGCN. The reason is that: our method takes the dynamic topology of traffic network into consideration while the existing methods donâĂŹt. As a result, our method can capture the propagation of traffic trends better. Furthermore, our method could avoid error propagation as in DCRNN.

G. EXPERIMENTAL RESULTS ANALYSIS

The reasons that our method achieves new state-of-the-art are from the following aspects. Compared with traditional methods, our deep model has larger capacity to describe the complex data dependency in traffic network. Second, our method takes the dynamic topology of traffic network into consideration while the existing methods donâĂŹt. As a result, our method can capture the propagation of traffic trends better. Finally, our network is also carefully designed for traffic prediction. The two-step prediction scheme breaks long-term predictions into two short-term predictions and makes the predictions easier.

VI. CONCLUSION AND FUTURE WORK

In this paper, we propose an effective and efficient framework DST-GCNN that can predict future traffic flows using surveillance videos. DST-GCNN is able to capture both the dynamics and complexity in traffic. The experiments indicate that our method outperforms other state-of-the-art methods. In the future, we plan to apply the framework to other traffic prediction tasks like pedestrian crowd prediction.

REFERENCES

[1] G. A. Davis, N. L. Nihan, M. M. Hamed, and L. N. Jacobson, "Adaptive forecasting of freeway traffic congestion," Transportation Research Record, no. 1287, 1990.

[2] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results," Journal of Transportation Engineering-asce, vol. 129, no. 6, pp. 664–672, 2003.

[3] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Wang, "Traffic flow prediction with big data: A deep learning approach," IEEE Transactions on Intelligent Transportation Systems, vol. 16, no. 2, pp. 865–873, 2015.

[4] W. Huang, G. Song, and e. a. Hong, "Deep architecture for traffic flow prediction: Deep belief networks with multitask learning," IEEE Transactions on Intelligent Transportation Systems, vol. 15, no. 5, pp. 2191–2201, 2014.

[5] X. Dai, R. Fu, Y. Lin, L. Li, and F.-Y. Wang, "Deeptrend: A deep hierarchical neural network for traffic flow prediction," arXiv preprint arXiv:1707.03213, 2017.

[6] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," national conference on artificial intelligence, pp. 1655–1661, 2016.

[7] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Graph convolutional recurrent neural network: Data-driven traffic forecasting," arXiv, 2017.

[8] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional neural network: A deep learning framework for traffic forecasting," arXiv preprint arXiv:1709.04875, 2017.

[9] H. K. Larry, "EventâĂŤbased shortâĂŤterm traffic flow prediction model," Transportation Research Record, vol. 1510, pp. 125–143, 1995.

[10] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Short-term traffic forecasting: Where we are and where weâĂŹre going," Transportation Research Part C: Emerging Technologies, vol. 43, pp. 3–19, 2014.

[11] M. X. Hoang, Y. Zheng, and A. K. Singh, "Fccf: forecasting citywide crowd flows based on big data," in Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. ACM, 2016, p. 6.

[12] X. Jin, Y. Zhang, and D. Yao, "Simultaneously prediction of network traffic flow based on pca-svr," Advances in Neural Networks–ISNN 2007, pp. 1022–1031, 2007.

[13] G. Leshem and Y. Ritov, "Traffic flow prediction using adaboost algorithm with random forests as a weak learner," in Proceedings of World Academy of Science, Engineering and Technology, vol. 19, 2007, pp. 193–198.

[14] Y. Wu and H. Tan, "Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework," arXiv preprint arXiv:1612.01022, 2016.

[15] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," arXiv preprint arXiv:1707.01926, 2017.

[16] H. V. Jagadish, J. Gehrke, A. Labrinidis, Y. Papakonstantinou, J. M. Patel, R. Ramakrishnan, and C. Shahabi, "Big data and its technical challenges," Communications of The ACM, vol. 57, no. 7, pp. 86–94, 2014.

[17] W. Liu, D. Anguelov, D. Erhan, and e. a. Szegedy, "Ssd: Single shot multibox detector," in ECCV. Springer, 2016, pp. 21–37.

[18] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," TPAMI, vol. 37, no. 3, pp. 583–596, 2015.

[19] A. G. Howard, M. Zhu, B. Chen, and e. a. Kalenichenko, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.

[20] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," arXiv preprint arXiv:1506.05163, 2015.

[21] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in Advances in Neural Information Processing Systems, 2016, pp. 3844–3852.

[22] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," Applied and Computational Harmonic Analysis, vol. 30, no. 2, pp. 129–150, 2011.

[23] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-rnn: Deep learning on spatio-temporal graphs," in CVPR, 2016, pp. 5308–5317.

[24] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in Advances in neural information processing systems, 2014, pp. 3104–3112.

[25] M. X. Hoang, Y. Zheng, and A. K. Singh, "Fccf: Forecasting citywide crowd flows based on big data," in Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ser. SIGSPACIAL âĂŹ16. New York, NY, USA: Association for Computing Machinery, 2016. [Online]. Available: https://doi.org/10.1145/2996913.2996934

· · ·