IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 5, 2019, accepted September 23, 2019, date of publication September 26, 2019,
date of current version October 8, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2943890

Densely Connected Convolutional Networks With Attention LSTM for Crowd Flows Prediction

WEI LI[1], WEI TAO[1], JUNYANG QIU[2], XIN LIU[1], XINGYU ZHOU[3], AND ZHISONG PAN[1]
[1]College of Command and Control Engineering, Army Engineering University of PLA, Nanjing 210007, China
[2]School of Information and Technology, Deakin University, Geelong, VIC 3217, Australia
[3]College of Communication Engineering, Army Engineering University of PLA, Nanjing 210007, China

Corresponding author: Zhisong Pan (hotpzs@hotmail.com)

This work was supported by the National Natural Science Foundation of China under Grant 61473149.

ABSTRACT With the rapid progress of urbanization, predicting citywide crowd flows has become increasingly significant in many fields, such as traffic management and public security. However, influenced by the complex spatiotemporal relations in raw data and other factors, such as events and weather, obtaining a precise prediction is challenging. Some previous works attempted to address this problem using various ways, such as autoregressive integrated moving average, vector auto-regression and some deep learning models. However, seldom can these methods comprehensively capture the spatiotemporal correlations. In this paper, we propose a novel spatio-temporal prediction model that is based on densely connected convolutional networks and attention long short-term memory (ST-DCCNAL), to simultaneously predict the inflow and outflow of the crowds in regions divided within a specific city. The ST-DCCNAL model consists of three parts: spatial part, external factors part and temporal part. In the spatial part, we employ densely connected convolutional networks to extract spatial characteristics at different levels. The external factors part utilizes a fully connected network to extract features from auxiliary information. In the last part, an attention-based long short-term memory module is leveraged to capture the temporal pattern. To demonstrate the practicality and effectiveness of the proposed model, we evaluate it using two separate real-world datasets of taxis in Beijing and bikes in New York. The experimental results confirm that the performance of our model is better than that of other baseline methods.

INDEX TERMS Data mining, spatiotemporal modeling, crowd flow prediction, densely connected convolutional network, long short-term memory, attention mechanism.

I. INTRODUCTION

As a typical spatiotemporal forecast problem, predicting the crowd flows in a city based on big data has a significant role in many fields such as traffic management [1], risk assessment [2], and public security [3]. The problem is aimed at acquiring an accurate prediction of crowd flows based on historical data. For instance, in the traffic management field, knowing the exact crowd number for each road in a whole city can solve traffic jams and congestion. A traffic management department can allocate traffic resources more appropriately depending on the crowd flow or issue appropriate traffic warnings. During holidays or events, people gather in a crowded and chaotic way, and prone to stampede accidents. Obtaining the crowd flows beforehand can ensure public safety by evacuating people, sending warnings to take precautions in advance.

Three difficulties arise in obtaining precise predictions of citywide crowd flows. 1) Complex spatial correlations: the correlations among neighbouring regions are involved. Adjacent regions are more similar and relevant than distant region according to the first law of geography. However, this kind of description is not comprehensive. For instance, in a city, regions with similar functions, such as shopping areas, tend to have a similar trend of crowd flows, while a long geographical distance may exist between these areas. 2) Dynamic temporal patterns: the movements of crowds are mostly periodic. People gather during peak hours when they travel to and from work, which produces a comparatively higher density of crowds in the morning and nightfall during workdays. However, temporal patterns are not strictly periodic. Some perturbations may occur from one period to

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

another period. 3) Other external factors: external factors such as meteorology and events have a considerable role in crowd flows prediction. For example, during a vocal concert or an evening party, people tend to gather together. Thus, the crowd flows increase on these days. The impact of various factors on the crowd flows becomes problematic.

In this paper, we focus on the citywide crowd flow prediction problem [4] and consider two types of crowd flows (inflow and outflow). Inflow denotes the total number of crowd numbers that enter a specific region during a given time interval, and the outflow is the reverse condition. As both of these two flows reflects the changes in crowd flow, controlling their details is significant. We can obtain the crowd flow data from the volume of vehicles, and pedestrians and mobile phone signals. Detailed definitions of the crowd flows are provided in section II.

Traditional spatiotemporal prediction methods often consider crowd flows prediction as a series of time series prediction problems and disregard the spatial correlation among these regions. Recently, some deep learning-based methods treat crowd flows as an image and apply simple convolutional neural networks (CNNs) to perform feature extracting. However, these methods only generate the final high-level feature and the uncorrelated regions may hinder the prediction performance. To learn the temporal patterns, some methods directly stack a series of images in chronological order [4], [5]. Yu *et al.* [6] utilize a long short-term memory (LSTM) [7] framework with an autoencoder while disregarding the spatial correlations. Yao *et al.* [8] concatenate an attention-based LSTM with a low-level LSTM to learn temporal patterns from features extracted by a CNN, but it is relatively complex.

To address the previously mentioned problem, we propose a hybrid model named ST-DCCNAL. In this model, we exploit the DenseNets [9] module to capture the spatial correlation. The DenseNets connects each CNN layer to every other layer in a feed-forward fashion. Thus, it is capable of enhancing feature reuse and strengthening feature propagation. In addition to the spatial information, other external features are formalized to address suddenness. Regarding the temporality, an attention-based mechanism is applied to assign different levels of attention to features according to their temporal closeness towards the predicting target. The features are then sent into an LSTM to discover the underlying temporal dependencies in the data. By combining these techniques and mechanisms, the proposed model can better capture spatiotemporal characteristics in the data, and fully utilize other relevant information. In this way, traffic flows can be precisely predicted, which plays a crucial role in intelligent urban management.

Our proposed model is validated via two real-world datasets (BikeNYC and TaxiBJ) including bike-sharing data for New York City in 2014 and taxi data for Beijing in 2013-2016. The total experimental results verify the superior performance of our model over other comparative models. The major contributions of this paper are summarized as follows:

- We apply a densely connected convolutional network to model spatial characteristics of regions concerning their neighbours on different levels. An attention LSTM mechanism is also employed to learn the periodic and dynamic temporality.
- We fully utilize auxiliary information such as weather and events information to improve the process of handling various unexpected circumstances.
- We conduct extensive experiments on two real-world trajectory datasets. The results demonstrate that our approach outperforms the six competing baseline models.

The remainder of this paper is structured as follows: we introduce some related work in section II. In section III, we define the problem and describe the techniques used in this paper. The detailed architecture of ST-DCCNAL model is described in section IV. We present the experimental results and analyze them in section V. In section VI, we conclude the paper and discuss prospects for future work.

II. RELATED WORK

The exploration of spatiotemporal sequence forecasts is a crucial component of spatiotemporal data mining. As a typical spatiotemporal prediction problem, crowd flows prediction problems, such as traffic speed prediction problems [10], are becoming increasingly important due to the construction of smart cities [11]. In this section, we briefly review related work on spatiotemporal prediction problems.

Previous work on the problem focuses on time series prediction. Regarding the spatiotemporal problem as a collection of many independent time series prediction problems, numerous traditional time series prediction methods can be applied to this field. For instance, autoregressive integrated moving average (ARIMA) methods have been extensively employed in traffic prediction problems [12]–[14]. Other conventional machine learning methods are utilized. Sun *et al.* [15] exploit a constructed Bayesian network to employ adjacent road links to analyse the impact of the cause node (data utilized for forecasting) to the effect node (data to be forecasted). Chen *et al.* [16] apply a Markov random field to identify traffic congestion locations to solve the uncertainty and low resolution of geographic locations. A fuzzy Bayesian network [17] is also employed in the spatiotemporal prediction field. However, these methods fail to capture the complicated and dynamic spatiotemporal correlations in the data.

Recently, deep learning methods have gained numerous successes in many fields, such as computer vision [18], speech recognition [19] and natural language processing [20]. CNN and LSTM have achieved success in the image processing area and sequence prediction area, respectively. In the trend of spatiotemporal prediction methods, an increasing number of people combine CNN, LSTM and other modules, such as a deep residual network (ResNet) [5], [21], an attention mechanism [22], [23], graph convolutional networks (GCN) [24], [25], and a deep belief network [26] to

construct a hybrid model to capture intricate spatiotemporal correlations. Khodayar and Wang [27] build a scalable graph convolutional deep learning architecture (GCDLA) to solve the wind speed forecasting problem. The GCDLA effectively processes noise while capturing the spatiotemporal features. He *et al.* [28] proposes the STANN model, which contains an encoder-decoder architecture with the attentive mechanism. The model is indicated to be active in traffic speed prediction. In literature [29], a sequential graph neural network is invented to replace the CNN in the traffic flow prediction problem. Concentrating on the connectivity of congestion road segments, rather than their spatial proximity, the model gains a better extraction of spatial features. Zang *et al.* [30] use a deep generative network that is based on residual deconvolution to solve the long-term traffic flow prediction problem. In addition to directly splicing the CNN module and the LSTM module, the two models can be merged into one module that is named Convolutional LSTM (ConvLSTM) [31]. Le Nguyen *et al.* [32] utilize a ConvLSTM-based architecture to solve the traffic matrix prediction problem and adequately model the spatiotemporal patterns. Due to the vast memory usage and computation cost of LSTM module, Ziru Xu *et al.* [33] invent the novel model PredCNN, which is an entirely CNN-based architecture. Using a multiplicative cascade unit that provides relative correlations from previous correlations, the PredCNN model is capable of predicting the future image without any recurrent chain structures.

The performance of the majority of these methods is better than traditional methods, and these methods are rapidly evolving. In these studies, the spatial correlations among regions are based on one perspective and disregard the different importance of each time interval. Wu *et al.* [34] propose a DNN based traffic flow prediction model (DNN-BTF) which combines CNN, LSTM, and an attention mechanism. However, our model exceeds DNN-BTF's capability of capturing spatial relationships.

Our proposed model explicitly handles the spatial relations in multiple different scales of convolutional layers, and the dynamic temporality is depicted via an attention LSTM module.

III. PRELIMINARIES

In this section, we propose the problem definition and discuss some basics about the attention mechanism and LSTM module in our model.

A. PROBLEM DEFINITION

Based on previous studies [5], we split the whole city into a $P \times Q$ grid map with n regions, where $n = P \times Q$ and a grid represents a region. Measurements for spatiotemporal prediction problems are diversiform, including air quality [35], weather, taxi order, and bike rent/return. Here, we use the crowd flows (including inflow and outflow) as the object of our study. The whole period is divided into K equal time intervals, and the crowd flows are recorded in the form of the collection of trajectories P. For the grid (p, q) at the k^{th} time interval, the inflow and outflow of this region can be defined as:

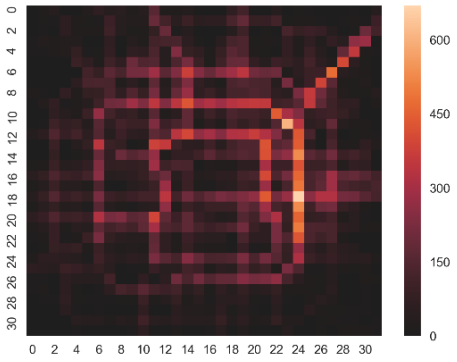


FIGURE 1. Inflow heat map in Beijing.

$$x_k^{in,p,q} = \sum_{T_{r_k} \in P} |\{i > 1 \mid g_{i-1} \notin (p,q), g_i \in (p,q)\}|, \quad (3.1)$$

$$x_k^{out,p,q} = \sum_{T_{r_k} \in P} |\{i \geq 1 \mid g_i \in (p,q), g_{i+1} \notin (p,q)\}|, \quad (3.2)$$

where g_i denotes a geographic coordinate, $g_i \in (p, q)$ indicates that the point g_i lies in region (p, q); $T_{r_k}: g_1 \rightarrow g_2 \rightarrow \cdots \rightarrow g_k$ represents the trajectory at the time interval $k (k = 1, 2, \ldots, K)$; and function $|\cdot|$ denotes the cardinality of a set.

If treating the grid map as an image with length P and width Q, the inflow and outflow of all regions at the k^{th} time interval can be denoted as the two-channel image $X_k \in R^{2 \times P \times Q}$ where $(X_k)_{0,p,q} = x_k^{in,p,q}$ and $(X_k)_{1,p,q} = x_k^{out,p,q}$. As shown in Figure 1, the bar on the right side denotes the relationship between the colour and the number of people in the flow. The horizontal axis and vertical axis are used to determine a specific coordinate.

With these notations, we can define the crowd flows prediction problem as follows:

Definition 1 (Crowd Flows Prediction): Given the history crowd flows data $\{X_t | t = 1, 2, \ldots, K\}$, *predict* X_{K+1}.

B. ATTENTION MECHANISM

As one of the most influential ideas in the deep learning community, the attention mechanism aims to overcome the drawback that the autoencoder structure satisfies a loss of information due to the fixed length of the middle vector when the length of the input sequence is relatively long. Focusing on certain parts of the input, the attention mechanism was initially designed for the seq2seq model in the Natural Language Process (NLP) and then quickly applied to other fields. The output of the attention mechanism can be written as:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad (3.3)$$

where $U = XW_U, U \in \{Q, K, V\}$ and X is the input, W_U are learnable matrices, d_k denotes the dimension of keys, K^T is the transpose of the matrix K. $softmax(\cdot)$ is an activation

function that is defined as $softmax\,(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$ where x_i is the i^{th} dimension of the N-dimensional vector $x(i, j = 1, 2, \ldots, N)$.

C. LONG SHORT-TERM MEMORY

In traditional sequence prediction models, a simple recurrent neural network (RNN) suffers from the gradient vanishing or exploding problem when calculating the back-propagation gradient if the layers of the RNN are deep. Therefore, RNN sometimes fails to capture the long-term dependency in a sequence, otherwise RNN should be able to do in theory. Hence, Long short-term memory is invented to solve this problem by recursively applying a transition function to the hidden state of the input.

LSTM retains a cell state C_t in the time interval t to stably learn sequential correlations. LSTM utilizes three gates—input gate, forget gate, and output gate—to control the information flow. In each time interval, LSTM considers C_{t-1}, h_{t-1} and x_t as an input, and the input gate i_t decides whether the previous information (h_{t-1} and x_t) is passed to the cell state. If the forget gate f_t is activated, the network will forget the previous memory cell C_{t-1}. The output gate o_t controls the output of the memory cell. The whole process of the LSTM unit is formulated as:

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right), \tag{3.4}$$
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right), \tag{3.5}$$
$$\hat{C}_t = tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right), \tag{3.6}$$
$$o_t = \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_o\right), \tag{3.7}$$
$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t, \tag{3.8}$$
$$h_t = o_t * \tanh\left(C_t\right), \tag{3.9}$$

where h_t denotes the hidden state of the LSTM unit at the time interval t, $*$ represents the element-wise multiply operation, tanh is the hyperbolic tanh function, and σ is the sigmoid activation function. W_v, $b_v(v \in \{f, i, C, o\})$ are parameters to be learnt.

The structure of LSTM units is shown in Figure 2.

IV. ST-DCCNAL MODEL

In this section, we provide the implementation details for our proposed ST-DCCNAL model. Figure 3 shows the main architecture of the model. The implementation of the three components in the ST-DCCNAL model, spatial part, external feature part and temporal part of the model are described.

A. OVERVIEW OF ST-DCCNAL

Treating the crowd flows in every region of the city at the time interval t as an image of shape $(2, P, Q)$, some techniques that are extensively employed in computer vision areas, such as CNN, can be employed to address the picture. First, we utilize a densely connected convolutional network to extract the spatial features. The purpose of the DenseNet architecture is to assist the CNN module to better capture spatial information as it can convey information between arbitrary CNN layers. When the spatial patterns have been obtained, they are reshaped into a vector and then fed into a fully-connected (FC) layer. The output of the FC layer is concatenated with the extract external features, which are transformed into the same shape. Thus, we can obtain the global feature of the crowd flows for the whole city at time interval t. To better obtain the temporal correlations of crowd flows at different time intervals, an attention-based LSTM is applied to handle global feature series. The generated spatiotemporal features are reshaped and fed into an activation function to obtain the prediction. An overview of the model is shown in Figure 3.

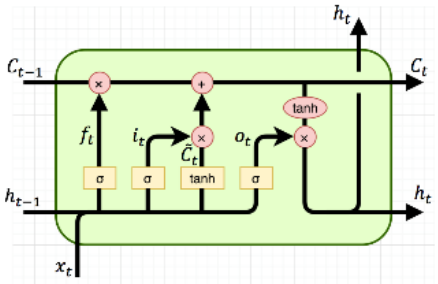


FIGURE 2. The structure of the LSTM unit.

B. SPATIAL PART

A big city can be divided into many regions corresponding to different spatial locations. We intuitively know that the crowd flows in nearby regions may affect each other. Thus, this effect can be effectively handled by a CNN, which has shown its powerful ability to hierarchically capture the spatial structural information. For instance, a crowd may flow into a specific region from its adjacent regions. Due to the development of various means of transportation such as bus and subway, correlations exist among distant regions. As we previously mentioned, including regions with weak correlations, predicting a target region hinders the performance and the central feature of a CNN are wasted. To address these issues, we leverage a DenseNet module to capture the spatial correlations among all regions. Our idea is motivated by Huang's work [9], during which the module retains a connection between each convolutional layer.

As shown in Figure 4, at the time interval i, we treat all regions as an image of the shape $(2, P, Q)$ and the channels denote the inflow and outflow during this time interval. We leverage the same padding to maintain a constant width and height. We implement the DenseNet structure by transmitting all outputs of the $i-1$ layers to the i^{th} layer. As a result, we obtain an image as the tensor (has two channels) X_t for each time interval t, $X_t \in R^{2\times P\times Q}$. Considering X_t as the input X_0, we feed it into K convolutional layers. The transformation at each layer k is defined as:

$$X_k = \sigma\left(\left(\sum\nolimits_{l=0}^{l=k-1} X_l\right) * W_t^{(k)} + b_t^{(k)}\right), \tag{4.1}$$

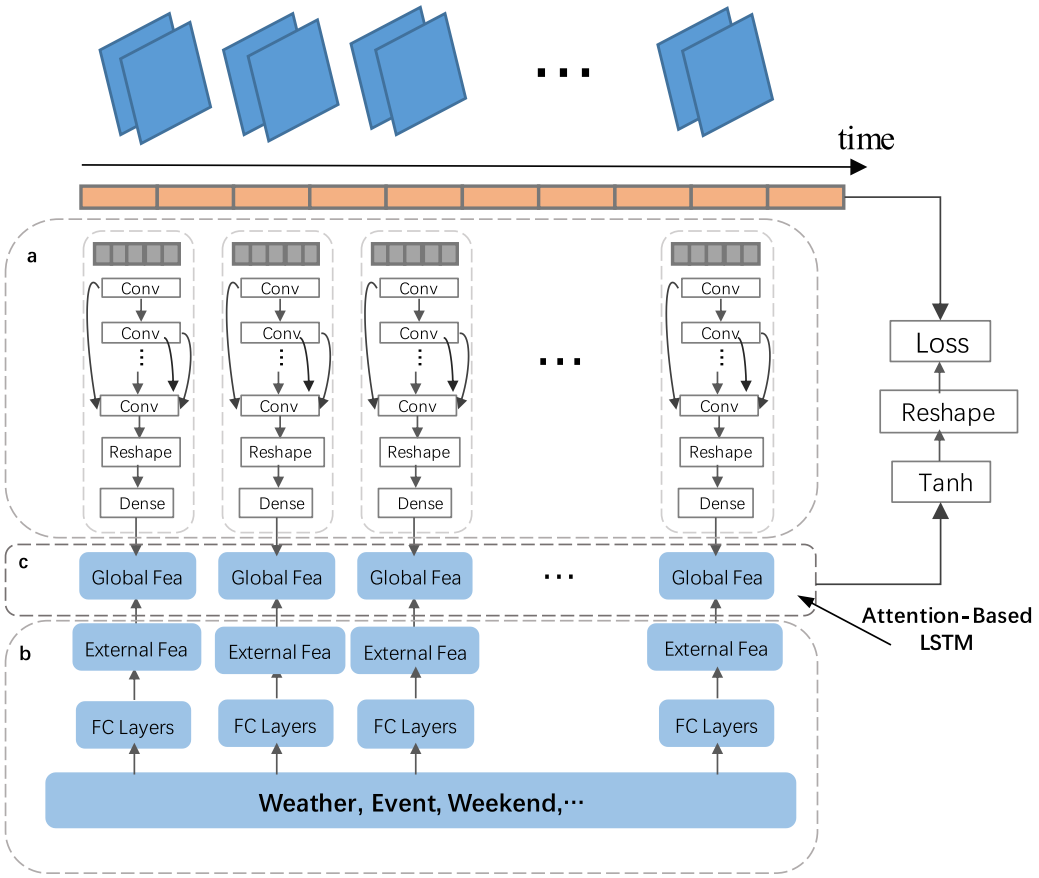


FIGURE 3. Architecture of ST-DCCNAL model. (a). The spatial part employs a DenseNet module to capture the spatial dependency among regions. (b). The external feature part extracts external features from auxiliary information, such as weather, event, and weekend information using FC layers. (c). The temporal part utilizes an attention-based LSTM module, the inputs of which are the concatenations of part (a) and part (b), to learn the temporal patterns. An activation layer and a reshape layer are used to generate predictions. Conv denotes CNN; Fea denotes feature, and FC represents fully-connected.

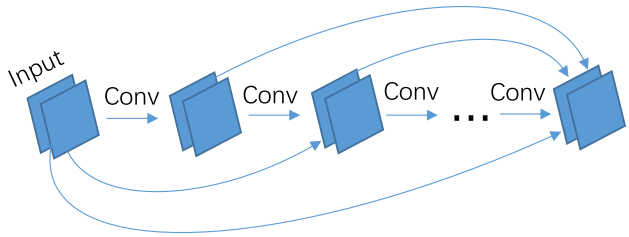


FIGURE 4. Structure of the densely connected convolutional network.

where $*$ denotes the convolutional operation, and $\sigma(.)$ is an activation function. In this paper, we use the rectifier function as the activation function, i.e., $\sigma(x) = max(0, x)$. $W_t^{(k)}$ and $b_t^{(k)}$ are two sets of parameters in the k^{th} layer to be trained.

The $\sum(.)$ operation denotes concatenating the input tensors by the first dimension (concatenate the channel dimension of images).

After K densely connected convolutional layers, we reshape the output $X_t \in R^{2\times P\times Q}$ into the feature vector $J_t \in R^{2PQ}$ for all regions at time interval t. To reduce the feature dimension, a dense layer is leveraged to generate the final spatial feature S_t which can be written as

$$S_t = \sigma\left(J_t * W_t^{sf} + b_t^{sf}\right), \tag{4.2}$$

where W_t^{sf} and b_t^{sf} are two learnable parameter sets at time interval t. For each time interval t, we obtain $S_t \in R^{dim}$ for all regions where dim denotes the dimension of the output of the DenseNets module.

C. EXTERNAL FEATURE PART AND FUSION

This part is an optional part depending on whether the original data contains auxiliary information such as weather, holiday, and event information. External features can provide global information which is always favorable for crowd flows prediction. To transform the auxiliary information into functional external features, we use a one-hot encoding mechanism to encode features that are nonnumeric. Information about a numeric class is normalized and then concatenated with the former to form the final external feature E_t at time interval t.

To generate the fused feature F_t of all regions at time interval t, the spatial feature S_t must be fused with the external

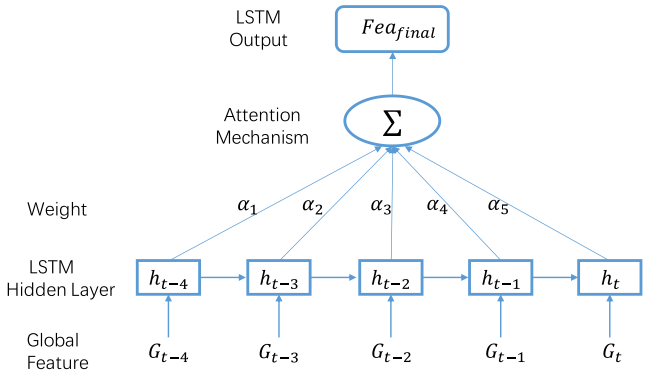


FIGURE 5. Sample of attention-based LSTM module.

feature E_t. The fusion process can be defined as:

$$F_t = tanh(W_t^{S\_fuse} \cdot S_t + W_t^{E\_fuse} \cdot E_t + b_t^{fuse}), \quad (4.3)$$

where $W_t^{S\_fuse}$, $W_t^{E\_fuse}$ and b_t^{fuse} are learnable parameters.

Considering that the model may overfit, we exploit a dropout layer after F_t to produce the eventual global feature G_t. For all regions at each time interval t, we obtain the global feature G_t.

D. TEMPORAL PART

An attention mechanism is, to some extent, motivated by how we pay visual attention to different regions of an image or words in one sentence. Introducing this mechanism into the LSTM network simplifies the selection of the inputs of previous layers that are crucial for each subsequent step. Treating the global feature G_t at each time interval t as the input, Figure 5 exhibits an attention-based LSTM module in which the length of the input sequence equals 5. We address the perturbation of the temporal patterns. We select L time intervals and leverage it to predict the next crowd flows. Combining the previous introduction of attention and LSTM, the extracted spatiotemporal feature ST in L selected time intervals is expressed as:

$$ST_{t+1} = f(ST_t, G_t), \quad (4.4)$$

where $f(\cdot)$ denotes an LSTM network, and G_t is the global feature at time interval t. The final feature by the attention mechanism is defined as:

$$Fea_{final} = \sum_{j=1}^{j=L} a_{tj} h_j, \quad (4.5)$$

where weight a_{tj} measures the importance of the time interval $j \in \{1, 2, \ldots, L$, h_j denotes the hidden state in the LSTM module at time interval j. The importance value a_{tj} is derived by comparing the learned spatiotemporal representation with a previous hidden state. The value is defined as follows:

$$a_{tj} = softmax\left(score\left(ST_j, h_j\right)\right). \quad (4.6)$$

In this formula, the $score(\cdot)$ function is regarded as a content-based function, which is defined as:

$$score\left(ST_j, h_j\right) = V^T \cdot \tanh\left(W_A \cdot h_j + U_A \cdot S_{t-1} + b_A\right), \quad (4.7)$$

Algorithm 1 ST-DCCNAL Training Algorithm

Input: Input : Historical observations:
$\{X_1, X_2, \ldots, X_T\}$;
external features: $\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_T\}$;
sequence length: L.
Output: Output : ST-DCCNAL model.
// generate samples from historical data
1. $\mathcal{D}_{sample} \leftarrow \emptyset$
2. **for** $t \in \{1, T - L\}$ **do**
3. put the following training sample
4. $(\{(X_t, \varepsilon_t), (X_{t+1}, \varepsilon_{t+1}), \ldots, (X_{t+L-1}, \varepsilon_{t+L-1})\}$, X_{t+L}) into $\mathcal{D}_{sample}$
5. **end**
6. divide $\mathcal{D}_{sample}$ into $\mathcal{D}_{train}$ and $\mathcal{D}_{test}$
// train the model
7. initialize all the parameters θ in ST-DCCNAL
8. **repeat**
9. randomly choose a batch of samples $\mathcal{D}_{batch}$ from $\mathcal{D}_{train}$
10. find θ by minimizing the objective (3.8) with $\mathcal{D}_{batch}$
11. **until** stopping criteria is met
12. output the learned ST-DCCNAL model

where W_A, U_A, and b_A are learnable parameters, and V^T is the transpose of V.

E. LOSS FUNCTION

We reshape the final spatiotemporal feature Fea_{final} and feed it into an activation function to generate the prediction $\hat{y}_i$ of the crowd inflow and outflow. Let y_i be the real crowd flows in all regions. The loss function is defined as:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left(y_i - \hat{y}_i\right)^2, \quad (4.8)$$

where θ denotes all parameters in the ST-DCCNAL model and N represents the number of samples.

Algorithm 1 outlines the training process of the ST-DCCNAL model. We construct the sample set D_{sample} from the historical observations (lines 2-5) and then divide D_{sample} into the training set D_{train} and the testing set D_{test}. The former is used to train the model, and the latter is utilized to test it. During each iteration, we optimize the objective function (4.8) on the chosen batch of training samples D_{batch} (lines 8-11).

V. EXPERIMENT

A. DATASETS DESCRIPTION

In this paper, we evaluate our proposed model using two large-scale real-world datasets from New York City and Beijing. Each dataset contains trajectories as follows:

•**BikeNYC:** The bike trajectory data were collected from New York City in 2014 [5], from 04/01/2014 to 09/30/2014 (183 days). Although it does not provide auxiliary information, the dataset contains inflow and outflow data and their

corresponding times. The time interval is set to 1 hour, and the whole city is divided into a 16×8 grid map. In the experiment, we choose the last 10 days as the testing data and the remaining days as the training data.

•**TaxiBJ:** The whole dataset contains inflow and outflow data in Beijing for four separate periods: July-Oct. 2013, March-June 2014, March-June 2015 and Nov. 2015-April 2016. The auxiliary information contains data on holidays and weather conditions, such as temperature, weather type, and wind speed. The city is divided into a 32×32 grid map while the time interval is half an hour, which produces the set of spatiotemporal images $X_t \in R^{2\times32\times32}$. We choose data from the last four weeks as testing data and select all data before this time as training data.

The two datasets can be easily accessed at https://www.jianguoyun.com/p/DesHv2UQs-HRBxi5gtYB.

B. EVALUATION METRIC

We use the rooted mean square error (RMSE) to evaluate the model performance, which is defined as:

$$RMSE = \sqrt{\frac{1}{M}\sum_{i=1}^{M}\left(v_i - \hat{v}_i\right)^2}, \tag{5.1}$$

where $\hat{v}_i$ and v_i are the predicted value and the ground truth respectively, and M denotes the total number of all predicted values.

C. BASELINES

In the experiment, we compare our model with four traditional models and four deep learning-based models as previously mentioned:

•**HA** [37]: The historical average (HA) method predicts the crowd flows by calculating the average values of the same region in a previous time interval. For instance, to predict the crowd flows in a specific region on Monday from 5:00 pm-6:00 pm, we can use all historical periods from 5:00 pm-6:00 pm in the same region on all Mondays as the actual data to generate the prediction. In this experiment, we employ historical data from the past seven days to obtain the current prediction. The HA model is easy and manageable but somewhat imprecise.

•**ARIMA** [38]: The autoregressive integrated moving average model (ARIMA), comprised of the autoregressive and moving average, is a typical model that is extensively applied in the time series prediction filed. In this experiment, we regard crowd flows prediction as a series of independent time series prediction problems.

•**SARIMA** [39]: A seasonal ARIMA (SARIMA) model is formed by including additional seasonal terms in the ARIMA models as previously mentioned. SARIMA is applied to handle the trend and seasonality of data. In this model, we use the seasonal difference for one or more times to eliminate the periodic variation. SARIMA appears to be more efficient than ARIMA model.

•**VAR** [40]: Vector auto-regression (VAR) is a model with a stochastic process that is used to estimate the dynamic relation of joint endogenous variables (multiple time series) and is capable of predicting spatiotemporal data. In this experiment, we regard the regions in the city as a vector. The inner correlations inside the vector and the temporality are learnt.

•**ST-ANN** [41]: This model extracts spatial (placing the region to be predicted in the centre of a 3×3 cell) and temporal (series of previous time intervals) features, which are fed into an artificial neural network. In this experiment, we set the number of previous time intervals to 8, and the number of units in the last dense layer is twice as the number of regions in a city.

•**DeepST** [4]: DeepST is an end-to-end prediction model for spatiotemporal data. Multi-source auxiliary information is utilized in this deep neural network-based model to capture spatiotemporal correlations. In DeepST, the data are sampled at three different time intervals, which represent the temporal closeness, period and seasonal trend, respectively. Moreover, the three sequences are subsequently fed into a convolutional neural network and fused.

•**ST-ResNet** [5]: ST-ResNet is a deep learning model that uses convolutional neural networks and residual networks to predict citywide crowd flows. Features from a different length of views (closeness, period and trend) are fused, and external information such as weather conditions and events information are leveraged to improve the performance. Via utilizing convolution-based residual networks, ST-ResNet adequately models the spatial patterns.

•**PredCNN** [33]: PredCNN is an entirely CNN-based architecture that models the dependencies between the next frame and the sequential video inputs. PredCNN was initially invented to handle the video prediction problem, which is an essential topic in spatiotemporal learning. Using a multiplicative cascade unit that provides relative correlations from previous correlations, the PredCNN model is capable of predicting a future image without any recurrent chain structures.

Subject to whether the spatial, temporal, and external features of data are considered, we conduct a brief comparison between our model and other baselines, as shown in Table 1.

TABLE 1. Comparison between our model and baselines.

Model	Temporal	Spatial	External
HA	√	×	×
ARIMA	√	×	×
STARIMA	√	×	×
VAR	√	×	×
ST-ANN	√	×	×
DeepST	√	√	√
ST-ResNet	√	√	√
PredCNN	√	√	√
ST-DCCNAL	√	√	√

D. PREPROCESSING AND PARAMETERS

For all regions, we leverage the Max-Min scaler to normalize crowd flow values that are nonnegative numbers to [0, 1]. For auxiliary information, we exploit the one-hot encoding

method to transform the discrete features (e.g., weather, and holidays) and use Max-Min normalization to scale continuous features such as wind speed and temperature to [0, 1]. The linear transformation of the original data is shown as:

$$x_i^* = \frac{x_i - x_{min}}{x_{max} - x_{min}}, \tag{5.2}$$

where x_i is a sample from the original data; x_{max} and x_{min} denote the maximum values and minimum values in the data, respectively; x_i^* is the transformed data. When we obtain predictions from the model, we re-scale them to generate the real predicted values.

We run all experiments on a cluster with eight NVIDIA 1080Ti GPUs. The python3.5 environments with TensorFlow and Keras [42] are used to build our models. The DenseNets part contains 3 layers of convolution, and 16 filters of size $(3, 3)$ in each convolutional layer. Batch normalization is employed in the CNN. For the temporal part, the sequence length L for LSTM are set to 12, and the units in the LSTM layer is set to 256. After the dense layer, we add a dropout layer with a dropout rate of 0.5 to maintain the generalization capabilities of our model. We set the batch size to 64, and use Adam [43] with a learning rate of 0.001 to optimize our model; The decay of the learning rate is 0.001. In the training process, we choose 90% data to train and the left 10% as the validation set. We set the maximum training epoch to 300, and apply early stopping with a patience of 20 to prevent the model from overfitting.

E. PERFORMANCE COMPARISON WITH BASELINES

Table 2 shows the performance of the proposed method compared with other competing baselines. ST-DCCNAL achieves the lowest RMSE with BikeNYC (5.41) and TaxiBJ (15.84), which is a 14.53% improvement and 5.01% improvement, respectively, over the best performance in baseline models. We observe that the HA model shows a poor performance compared with other models (i.e., RMSE of 11.76 and 57.69 for BikeNYC and TaxiBJ, respectively), as it only relies on historical data while disregarding spatial correlations and other external information.

Efficiently exploiting more data and information, deep-learning-based models perform substantially better than conventional methods, such as HA, ARIMA, and VAR. As shown in Table 2, ST-ResNet and DeepST outperform ST-ANN. A potential reason is that ST-ResNet and DeepST utilize CNNs to capture spatial information and consider the temporal periodicity. PredCNN outperforms ST-ResNet, but does not perform as well as our model. Although the cascade multiplicative unit (CMU) in PredCNN can transmit the intermediate results of the previous time intervals, it loses some historical information.

Using DenseNet and attention LSTM to capture spatiotemporal patterns among regions, our proposed ST-DCCNAL model outperforms the previously mentioned methods.

TABLE 2. Comparison with different baselines on BikeNYC and TaxiBJ (to accelerate the experiments, some baseline results are cited from [5]).

Dataset	Model	RMSE
BikeNYC	HA	11.76
	ARIMA	10.07
	STARIMA	10.56
	VAR	9.92
	DeepST	7.43
	ST-ResNet	6.33
	PredCNN	5.61
	ST-DCCNAL	**5.41±0.10**
TaxiBJ	HA	57.69
	ARIMA	22.78
	STARIMA	26.88
	VAR	22.88
	ST-ANN	19.57
	DeepST	18.18
	ST-ResNet	16.69
	PredCNN	15.90
	ST-DCCNAL	**15.63±0.10**

F. PERFORMANCE COMPARISON WITH MODEL VARIANTS

We also study the effect of different components proposed in our model to confirm their effectiveness. As the model consists of three parts, we verify each part by deleting or replacing it to form relatively complete comparison results. Due to the limitations of dataset BikeNYC (does not contain any external supplementary information), we decide to conduct these experiments using dataset TaxiBJ. The model and its variants are described as follows:

•**CNN**: For this variant, we only use a simple CNNs with inputs to evaluate the CNNs' capability of extracting spatial correlations among regions and provide a baseline for other variants. To maintain a constant shape of the output image, we use the same padding mechanism.

•**CNN + Concatenate**: In this variant, the information in the middle CNN layers is efficiently utilized. The output of each convolutional layer is concatenated in the last convolutional layer; therefore, the hidden information can be properly reused.

•**DenseNets**: For this variant, normal CNNs are replaced with a DenseNets structure. We want to demonstrate that

DenseNets can model complex spatial relations better than normal CNNs.

•**DenseNets + LSTM**: This variant considers both spatial information and temporal information to generate the prediction. The input is fed into the DenseNets to extract the spatial relations, and then an LSTM is applied to extract the temporal pattern.

•**DenseNets + LSTM + Attention**: This variant differs from **DenseNets + LSTM** as we add an attention part before the LSTM module. The Attention mechanism helps the LSTM module to better capture the temporality.

TABLE 3. Comparison with different baselines for BikeNYC.

Method	RMSE
CNN	22.84
CNN + Concatenate	22.19
DenseNets	17.20
DenseNets+ LSTM	16.31
DenseNets + LSTM + Attention	15.93
ST-DCCNAL	15.63

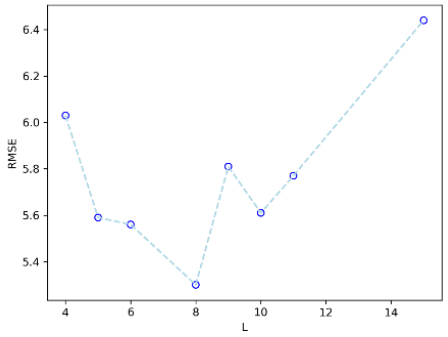


FIGURE 6. The effect of sequence length L on RMSE.

•**ST-DCCNAL**: Our proposed model, which combines DenseNets, LSTM, the attention mechanism and external feature, can combine the advantages of each module and obtain a better prediction.

Table 3 shows the performance of ST-DCCNAL and its variants for TaxiBJ. During the process of the spatial feature extraction, the DenseNets module achieves a lower RMSE (a reduction of 24.79% and 22.49% compared with CNN and CNN with Concatenate, respectively). The results demonstrate that the effectiveness of connecting all CNN layers. The DenseNets module can better capture spatial relations than normal CNN and Concatenating CNN. Furthermore, adding an LSTM module to DenseNets improves the performance of DenseNets, which confirms the effectiveness of the temporal part. As shown in Table 3, DenseNets with attention LSTM outperforms DenseNets with LSTM, which proves that the attention mechanism enables LSTM to better capture temporal patterns. When adding the external part to form our proposed model ST-DCCNAL, we discover that the prediction performance improves as well. Thus, the external feature from auxiliary information facilitates the prediction.

G. INFLUENCE OF SEQUENCE LENGTH OF ATTENTION LSTM

In this section, we conduct the parameter sensitivity analysis for the experiment to address the uncertainty of the data. We explore how the sequence length of LSTM in the temporal part influences the prediction performance. In the sensitivity analysis experiment, we adjust the value of the sequence length of the attention LSTM module and maintain constant hyperparameters. The value of the sequence length L varies in the range of {4, 5, 6, 8, 9, 10, 11, 15}, and we observe the RMSE of the predictions. Figure 6 shows the prediction RMSE with different sequence length L. The RMSE error initially decreases at first and then increases. The variance and the mean of the error are 0.1051 and 5.76, respectively. When the sequence length is 8 hours, the ST-DCCNAL model achieves the best performance. The RMSE decreases as the length increases, considering the temporal dependency. Furthermore, when the sequence length exceeds 10 hours, the performance gradually worsens. When the sequence length is longer, the model becomes considerably more complex and may overfit, which hinders the performance.

VI. CONCLUSION AND FUTURE WORK

In this paper, we propose a novel spatio-temporal prediction model that is based on densely connected convolutional networks and long short-term memory with Attention (ST-DCCNAL) for crowd flows prediction. Our approach exploits a DenseNet module to capture the spatial dependency and then fuse it with external features. An attention LSTM module is applied to extract temporal patterns. We innovatively combine these deep learning techniques to form a novel traffic flows prediction model. The proposed model is capable of extracting the deeply hidden complex spatiotemporal features. We evaluate our model using two real-world crowd flows datasets, and the experimental results demonstrate that the ST-DCCNAL model significantly outperforms several competing methods. We discover that the DenseNets module models the spatial relations better than norm CNNs and the attention mechanism improves the ability to capture the temporality of LSTM. Furthermore, the external feature from auxiliary information can improve the accuracy of prediction.

In the future, we plan to further investigate the dynamic correlations among regions using deep learning techniques such as graph neural networks (GNN) [24], [44]. In most real situations, the division of a city does not conform to the regular image and GNN can show and unearth more complex correlations than CNN. Thus, the GNN is an effective method and CNN should be replaced with GNN in this domain. As interpreting deep learning methods is often challenging, understanding what contributes to the improvement is essential. We plan to further interpret the model and obtain the corresponding actual interpretation of the prediction. A model that can be updated in real-time is needed to handle the changing situations in many real cases. Therefore, we introduce the online learning mechanism [45], [46] to spatiotemporal sequence prediction problems, which is capable of providing a real-time prediction and updating the model. Everyone can easily access all data and code at https://github.com/liweiowl/ST-DCCNAL.

REFERENCES

[1] G. Shen, C. Chen, Q. Pan, S. Shen, and Z. Liu, "Research on traffic speed prediction by temporal clustering analysis and convolutional neural network with deformable kernels (May, 2018)," *IEEE Access*, vol. 6, pp. 51756–51765, 2018.

[2] M. Zheng, T. Li, R. Zhu, J. Chen, Z. Ma, M. Tang, Z. Cui, and Z. Wang, "Traffic accident's severity prediction: A deep-learning approach-based CNN network," *IEEE Access*, vol. 7, pp. 39897–39910, 2019.

[3] M. X. Hoang, K. Yu, and A. K. Singh, "FCCF: Forecasting citywide crowd flows based on big data," in *Proc. 24th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, Nov. 2016, Art. no. 6.

[4] J. Zhang, Y. Zheng, D. Qi, R. Li, and X. Yi, "DNN-based prediction model for spatio-temporal data," in *Proc. 24th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, Oct./Nov. 2016, Art. no. 92.

[5] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proc. 31st AAAI Conf. Artif. Intell.*, Dec. 2017, pp. 1655–1661.

[6] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, and Y. Liu, "Deep learning: A generic approach for extreme condition traffic forecasting," in *Proc. 2017 SIAM Int. Conf. Data Mining. Soc. Ind. Appl. Math.*, Jun. 2017, pp. 777–785.

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[8] H. Yao, X. Tang, H. Wei, G. Zheng, and Z. Li, "Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 5668–5675.

[9] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 4700–4708.

[10] Y. Gu, W. Lu, L. Qin, M. Li, and Z. Shao, "Short-term prediction of lane-level traffic speeds: A fusion deep learning model," *Transp. Res. C, Emerg. Technol.*, vol. 106, pp. 1–16, Sep. 2019.

[11] J. Massana, C. Pous, L. Burgas, J. Melendez, and J. Colomer, "Identifying services for short-term load forecasting using data driven models in a Smart City platform," *Sustain. Cities Soc.*, vol. 28, pp. 108–117, Jan. 2017.

[12] M. Van Der Voort, M. Dougherty, and S. Watson, "Combining Kohonen maps with ARIMA time series models to forecast traffic flow," *Transp. Res. C, Emerg. Technol.*, vol. 4, no. 5, pp. 307–318, 1996.

[13] X. Li, G. Pan, Z. Wu, G. Qi S. Li, D. Zhang, W. Zhang, and Z. Wang, "Prediction of urban human mobility using large-scale taxi traces and its applications," *Frontiers Comput. Sci.*, vol. 6, no. 1, pp. 111–121, Feb. 2012.

[14] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi-passenger demand using streaming data," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1393–1402, Sep. 2013.

[15] S. Sun, C. Zhang, and G. Yu, "A Bayesian network approach to traffic flow forecasting," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 1, pp. 124–132, Mar. 2006.

[16] P.-T. Chen, F. Chen, and Z. Qian, "Road traffic congestion monitoring in social media with hinge-loss Markov random fields," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2014, pp. 80–89.

[17] M. Das and S. K. Ghosh, "FB-STEP: A fuzzy Bayesian network based data-driven framework for spatio-temporal prediction of climatological time series data," *Expert Syst. Appl.*, vol. 117, pp. 211–227, Mar. 2019.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[19] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 6645–6649.

[20] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1188–1196.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[23] 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.*, Apr. 2018, pp. 2588–2595.

[24] P. W. Battaglia *et al.*, "Relational inductive biases, deep learning, and graph networks," Jun. 2018, *arXiv:1806.01261*. [Online]. Available: https://arxiv.org/abs/1806.01261

[25] Geng, Xu, "Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, 2019, pp. 3656–3663.

[26] L. Li, L. Qin, X. Qu, J. Zhang, Y. Wang, and B. Ran, "Day-ahead traffic flow forecasting based on a deep belief network optimized by the multi-objective particle swarm algorithm," *Knowl.-Based Syst.*, vol. 172, pp. 1–14, May 2019.

[27] M. Khodayar and J. Wang, "Spatio-temporal graph deep neural network for short-term wind speed forecasting," *IEEE Trans. Sustain. Energy*, vol. 10, no. 2, pp. 670–681, Apr. 2019.

[28] Z. He, C.-Y. Chow, and J.-D. Zhang, "STANN: A spatio–temporal attentive neural network for traffic prediction," *IEEE Access*, vol. 7, pp. 4795–4806, 2018.

[29] Z. Xie, W. Lv, S. Huang, Z. Lu, B. Du, and R. Huang, "Sequential graph neural network for urban road traffic speed prediction," *IEEE Access*, to be published.

[30] D. Zang, Y. Fang, Z. Wei, K. Tang, and J. Cheng, "Traffic flow data prediction using residual deconvolution based deep generative network," *IEEE Access*, vol. 7, pp. 71311–71322, 2019.

[31] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 802–810.

[32] Van An Le, P. L. Nguyen, and Y. Ji, "Deep convolutional LSTM network-based traffic matrix prediction with partial information," in *Proc. IFIP/IEEE Symp. Integr. Netw. Service Manage. (IM)*, Apr. 2019, pp. 261–269.

[33] Z. Xu, Y. Wang, M. Long, J. Wang, and M. O. KLiss, "PredCNN: Predictive learning with cascade convolutions," in *Proc. IJCAI*, Jul. 2018, pp. 2940–2947.

[34] Y. Wu, H. Tan, L. Qin, B. Ran, and Z. Jiang, "A hybrid deep learning based traffic flow prediction method and its understanding," *Transp. Res. C, Emerg. Technol.*, vol. 90, pp. 166–180, May 2018.

[35] X. Yi, J. Zhang, Z. Wang, T. Li, and Y. Zheng, "Deep distributed fusion network for air quality prediction," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2018, pp. 965–973.

[36] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[37] J. Y. Campbell and S. B. Thompson, "Predicting excess stock returns out of sample: Can anything beat the historical average?" *Rev. Financial Stud.* vol. 21, no. 4, pp. 1509–1531, Jul. 2007.

[38] M. S. Ahmed and A. R. Cook, "Analysis of freeway traffic time-series data by using box-jenkins techniques," *Transp. Res. Rec.*, no. 722, pp. 1–9, 1979.

[39] M. Valipour, "Long-term runoff study using SARIMA and ARIMA models in the United States," *Meteorol. Appl.*, vol. 22, no. 3, pp. 592–598, Jul. 2015.

[40] S. R. Chandra and H. Al-Deek , "Predictions of freeway traffic speeds and volumes using vector autoregressive models," *J. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 53–72, 2009.

[41] W. Jiaqiu, D. Min, and C. Tao, *Time-Space Sequence Data Analysis and Modeling*. Beijing, China: Science Press, 2012.

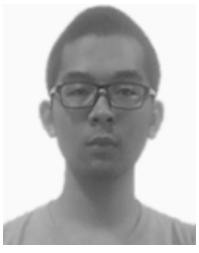
[42] F. Chollet. (2015). *Keras*. [Online]. Available: https://github.com/fchollet/keras

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

[44] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," Dec. 2018, *arXiv:1812.08434*. [Online]. Available: https://arxiv.org/abs/1812.08434

[45] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, pp. 2121–2159, Jul. 2011.

[46] S. Shalev-Shwartz, "Online learning and online convex optimization," *Found. Trends Mach.*, vol. 4, no. 2, pp. 107–194, Mar. 2012.

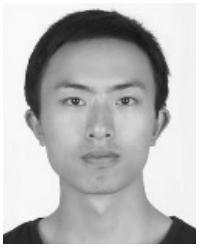


WEI LI received the bachelor's degree in computer science and technology from Peking University, in 2017. He is currently pursuing the master's degree in computer science and technology with the Army Engineering University of PLA, China. His main research interests include machine learning and spatiotemporal sequence prediction.

WEI TAO is currently pursuing the Ph.D. degree with the Army Engineering University of PLA, Nanjing, China. His main research interests include machine learning, optimization algorithms, and network security.

JUNYANG QIU is currently pursuing the Ph.D. degree with the School of Information Technology, Deakin University, Australia. His research interests include malware analysis and machine learning.

XIN LIU is currently pursuing the Ph.D. degree with the Army Engineering University of PLA, Nanjing, China. His main research interest includes machine learning.



XINGYU ZHOU received the M.S. degree in information and communication engineering from the PLA University of Science and Technology, Nanjing, China, in 2011. He is currently pursuing the Ph.D. degree with the Army Engineering University of PLA. His research interests include computer vision and pattern recognition.

ZHISONG PAN received the Diploma degree in computer science and technology from PLA Information Engineering University, China, in 1996, and the Ph.D. degree in computer science and technology from the Nanjing University of Aeronautics and Astronautics, in 2003. He is currently a Professor with the Army Engineering University of PLA. His main research interests include machine learning, pattern recognition, and network security.