Applied Soft Computing Journal 97 (2020) 106688

Contents lists available at ScienceDirect

Applied Soft Computing Journal

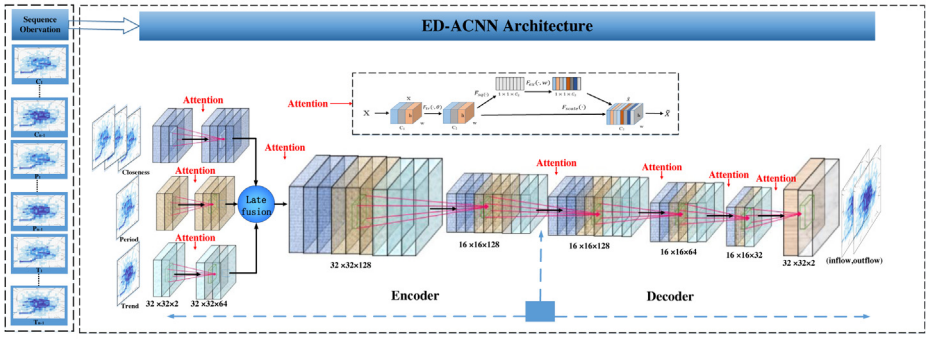
journal homepage: www.elsevier.com/locate/asoc

ED-ACNN: Novel attention convolutional neural network based on encoder–decoder framework for human traffic prediction

Bin Pu [a], Yuan Liu [a], Ningbo Zhu [a,*], Kenli Li [a], Keqin Li [a,b]

[a] College of Computer Science and Electronic Engineering, Hunan University, China
[b] Department of Computer Science, State University of New York, New Paltz, NY 12561, USA

GRAPHICAL ABSTRACT



ARTICLE INFO

Article history:
Received 16 October 2019
Received in revised form 12 August 2020
Accepted 26 August 2020
Available online 18 September 2020

Keywords:
Attention mechanism
Encoder–decoder framework
Spatio-temporal data mining
Traffic-flow prediction

ABSTRACT

Accurate human traffic prediction, as a vital component of an intelligent transportation system (ITS), can not only reduce traffic congestion and resource consumption, but also provide a foundation for other tasks, such as risk assessment and public safety. Owing to the rapid development of computing power, massive data storage, and parallelization, deep-learning techniques, especially convolutional neural networks (CNNs), have become a powerful tool for traffic-flow forecasting. However, most of these methods in the literature over-emphasize the accuracy of traffic-flow forecasting and ignore its efficiency. It is often beneficial to develop smaller models (e.g., fewer model parameters) to improve efficiency. In this work, taking into account the efficiency and accuracy of the prediction, a novel attention CNN based on an encoder–decoder framework, called ED-ACNN, is proposed. First, the convolutional layer is considered the coding layer to extract spatial and temporal correlations. Then, the deconvolution layer as a decoding layer is expertly designed to reconstruct the future traffic-flow image. Next, the attention mechanism is introduced into the proposed model to capture the correlation between the spatial traffic-flow images' channels. Finally, for the three characteristics of *closeness*, *period*, and *trend*, it is concluded that the *closeness* feature is the most significant for human traffic prediction in the proposed approach. An extensive experimental evaluation of two types of real-world crowd flow (Beijing and New York City) is presented, and the results show that the proposed method can be very competitive with state-of-the-art baselines.

© 2020 Elsevier B.V. All rights reserved.

1. Introduction

Along with urbanized progress, traffic-flow forecasting is becoming increasingly essential to traffic management, resource consumption, and public safety [1,2]. The increase in numbers of

* Corresponding author.
E-mail address: quietwave@hnu.edu.cn (N. Zhu).

https://doi.org/10.1016/j.asoc.2020.106688
1568-4946/© 2020 Elsevier B.V. All rights reserved.

vehicles and population can lead to traffic congestion, environmental pollution, and social security problems [3]. For example, there was a stampede on New Year's Eve at Shanghai's Bund in 2015 [4] and a dangerous stampede in mid-July of 2016 for the "Pokemon Go" game [5]. Once the ability to predict traffic flow in each region of a city is realized, similar events can be avoided. Research on traffic-volume forecasting began in the 1970s and has been ongoing for nearly 50 years. There are many known traditional traffic-flow forecasting methods reported in the literature, which can be roughly divided into the following three categories.

The first category is the parametric approach, which mainly uses the autoregressive integrated moving-average-based (ARIMA-based) methods or the Kalman-filtering models to solve the time series [6]. The ARIMA model is used to predict the short-term traffic flow of freeways [7], and its various [8–11] deformation methods have also been applied to traffic-flow prediction problems. Most Kalman-filtering models are designed to predict short-term traffic flow. For example, Guo et al. proposed an adaptive Kalman filter to implement the SARIMA+GARCH structure, which can provide improved adaptability when traffic is highly volatile [12].

The second category is the non-parametric methods, which usually better capture the uncertainty of traffic time series and complex nonlinearities, such as an artificial neural network (ANN), support vector regression (SVR), Bayesian networks, and K-nearest-neighbor (K-NN) [13–18]. ANN, as a typical non-parametric method, has been widely used in short-term traffic-flow prediction [13]. SVR has been successfully used to predict traffic conditions such as hourly flow, travel time, and prediction of short-term freeway traffic flow under both typical and atypical conditions [14,15]. Sun et al. designed a Bayesian network that is modeled by traffic flow among adjacent road links in a transportation network [16]. Davis et al. first proposed K-NN as a non-parametric regression approach for short-term freeway traffic forecasting, which may avoid several of the problems inherent in parametric forecasting approaches [18].

Given that the above methods have their own good performance for a specific period, researchers have turned their attention to combining predictors to predict the periods of multiple spans, especially for combinations of ANN and other algorithms. Zhang et al. suggested a hybrid method combining ARIMA and ANN models that uses the unique advantages of ARIMA and ANN models in linear and nonlinear modeling [19]. Moretti et al. presented a hybrid modeling method that combines an ANN and a statistical approach for forecasting hourly urban traffic-flow rates [20]. Some improved approaches are combined with other predictors, such as fuzzy theories [21] and population-based optimization methods [22,23].

Although these studies offer a variety of options with which to tackle traffic-flow forecasting problems, research in this area is far from mature. For example, most parametric approaches based on time series of linear architectures cannot capture nonlinear and spatial features. Thus, it is insufficient to rely on parametric models alone to predict traffic volume; that is, the ANN approaches have low prediction accuracy with the limited data and over-fitting with massive instances. Traditional shallow neural networks, such as [13], cannot obtain high-level abstract features. Although both SVR and Bayesian networks can solve problems like small samples and nonlinearities, their prediction accuracy is affected by a small amount of noise or random factors caused by the variability of traffic data. Thus, non-parametric methods are not the best for predicting traffic volume. The hybrid models are primarily intended to predict highway traffic with fewer data training models. However, they are challenging to use to extract advanced temporal and spatial features for large amounts of spatio-temporal data and complex urban environments. In addition, most existing methods predict flow on single- or multiple-road segments, rather than on citywide ones. This demonstrates that there are some regions where stampede events may occur that cannot be predicted, like the bunds and public parks [4] where stampede events may occur but are not predicted. In addition, most of the previous methods cannot adapt to massive spatio-temporal data. How to effectively predict crowd flow with exploding traffic data is emphasized in [24–26].

Deep learning (DL), as a new data-driven machine-learning method, provides new ideas for traffic prediction with exploding data. Owing to the development of new variants of CNNs and the advent of efficient parallel solvers optimized for modern GPUs, DL approaches have recently been successfully applied to many areas, such as speech recognition, pattern recognition, and computer vision [27]. DL approaches have also been found to be suitable for Big Data analysis [27]. The rapid developments of DL techniques and a large number of mobile location data acquisitions (e.g., population, vehicle, and devices) make it possible to build accurate models for a wide range of transportation applications, such as estimating traffic flow, anticipating the time of path, and forecasting crowd density.

As is well known, traffic flow is not a static variable, but is a highly real-time one. Real-time prediction is significant in traffic-volume forecasting, particularly in short-term prediction and human emergency behavior forecasting [28]. In other words, it requires forecasting to be highly efficient. On the one hand, compared with traditional ANN algorithms, fewer parameters are required for CNNs, which leads to a significant reduction in memory and an improvement in efficiency. On the other hand, one can implement CNN-based methods on modern GPU processors to increase the efficiency of processing large amounts of spatio-temporal data. At the same time, a CNN-based method can automatically extract the intermediate- and high-level abstractions obtained from a large amount of original spatio-temporal data (i.e., spatial traffic-flow images) to complete the current prediction task. In summary, the CNN-based models are suitable for current human traffic prediction. Recently, many scholars have presented several approaches based on CNNs [5,29–32].

Although CNN-based models have certain advantages in predicting tasks, they still exhibit the problem of reduced efficiency with increasing layers, especially for a large dimension of the input image, large size of the receptive field, and more fully connected layers. For instance, VGG-19 [33], as an image-classification model, contains 575 MB of parameters and most of its model parameters come from fully connected layers, which will have a low model efficiency compared with other models, such as GoogLeNet-v1 (53 MB of parameters) [34]. In addition, large model parameters are more likely to lead to overfitting and more data for training. Given the same level of accuracy, it is often beneficial to design smaller CNNs (i.e., CNNs with fewer model parameters), as discussed in [35]. However, most scholars only pay attention to accuracy in traffic-flow forecasting.

In this work, use of a simple model structure is recommended to ensure accuracy and efficiency. The proposed ED-ACNN is a compact model based on the encoder–decoder framework. To the best of our knowledge, it is the first time human traffic-flow has been predicted based on an encoder–decoder framework. At the same time, the attention block is introduced in ED-ACNN to capture the correlation of the images' channels. Experimental results show that ED-ACNN demonstrates good performance for human traffic prediction. The contributions of this work are the following.

- A novel attention CNN based on an encoder–decoder framework is proposed for forecasting the flow of crowds, which consists of encoder module, attention unit, and decoder component.

- ED-ACNN demonstrates the advantages of a more simple structure, multi-step forecasting, and higher instance-level precision by giving up all fully connected layers.
- The attributes of spatio-temporal data are summarized into three categories: *closeness*, *period*, and *trend*. It is concluded that the *closeness* attribute is the most important for crowd-flow forecasting in this work by analyzing a single traffic flow-property.
- The proposed method is evaluated on two types of crowd flow data in Beijing and New York City. The experimental results show the advantages of the proposed approach compared with well-known baselines.

The rest of this paper is arranged as follows. We introduce the related work in Section 2. The related problem statements and notations are defined in Section 3. We present the detail of the proposed work in Section 4. Section 5 demonstrates the experiment, and then a discussion is given based on the results. Finally, the conclusion and the direction of future work are organized in Section 6.

2. Related work

In this section, an overview research of traffic-volume prediction based on DL approaches is presented.

Deep learning: DL has achieved great success in many areas [27], such as speech recognition [36,37], computer vision [38,39], and natural language processing [40]. Recently, many researchers have used DL methods to predict traffic flow. A deep belief network (DBN) is used for traffic-flow prediction [41,42]. Huang et al. presented a deep architecture that consists of two parts for traffic-flow prediction, with a DBN at the bottom and a multitask regression layer at the top [42]. Lv et al. proposed a stacked autoencoder model, which is used to learn generic traffic-flow features that are the inherent spatial and temporal correlations [26]. Then, CNN-based approaches were designed for spatio-temporal prediction. Deng et al. explored spatio-temporal relations via deep CNNs, which can tolerate the incomplete traffic data [32]. Zhang et al. designed a DL-based prediction model for spatio-temporal data (DeepST), which consists of three parts: dependent instances, CNNs, and model fusions [29]. Then, Zhang et al. designed a deep spatio-temporal residual network (ST-ResNet), which learns on an aggregate output of the three residual NNs based on spatio-temporal data [5]. Although ST-ResNet achieves a state-of-the-art result, it comes at the cost of a large number of model parameters and a huge amount of computation, which limits its applications. Moreover, this approach does not take into account the dependencies between different feature maps. These studies have drawn significant academic interest. For example, Zonoozi et al. proposed a recurrent convolutional model (PCRN), which adapts the recurrent convolutional network to capture the spatio-temporal and fusion periodic representations [30]. Previous research efforts have been made using two-dimensional (2D) CNNs. Chen et al. designed a multiple 3D CNNfor city-wide vehicle-flow prediction, which learns the spatio-temporal correlation features jointly from low- to high-level layers for traffic data [31]. Video-sequence analysis and prediction have similarities with traffic flow, both of which have spatial and temporal correlations. In the next-frame prediction problem in video analysis, many approaches have achieved great success, such as a recurrent convolutional network for visual learning [43], ConvLSTM [44], and ConvGRU [45], which are better at extracting spatial regularities.

Encoder–decoder framework: This is mainly used for machine translation [46] and image segmentation [47], such as FCN [48] and unet [49]. It is proved by [49] that unet is a simple and effective method, and it is widely applied to biomedical image-segmentation tasks. The encoder–decoder framework also projects onto and maps back from feature-map space or continuous vector space [46], which means that the high- and hidden-level features can probably be extracted. ED-ACNN consists of a convolution layer and deconvolution layer that act as an encoder and decoder pair. The encoder maps the variable-length image sequences to the feature maps, and the decoder maps the feature-map representation back to the target image. The attention mechanism can assist the model focus more on information that contributes a lot to the output, and some attention mechanism networks are introduced in [50,51]. In this work, the appropriate flexible SEnet attention blocks [52] are added by setting weights on the multi-channel feature maps to capture the correlation between channels. Finally, the attributes (i.e., *closeness*, *period*, and *trend*) of traffic-flow characteristics are analyzed.

3. Problem statement

In this section, we mainly define some notations and problem statements. The factors involved with the traffic information of human traffic volume primarily comprises the region, spatial traffic flow image, inflow, outflow, spatial dependencies, temporal dependencies, late fusion, and human traffic prediction. They are defined as follows:

Definition 1 (*Region*)**.** Region of this kind is a grid area based on the real map, which can almost cover the citywide ones. The target area is divided into $I \times J$ grids. For example, we map Beijing into 32 $\times$ 32 regions, with 1 km $\times$ 1 km grid size each. Fig. 1(b) illustrates the schematic diagram for distributing the grid area.

Definition 2 (*Spatial Traffic Flow Image*)**.** We pre-defined geographic region during a fixed time interval (0.5 h or 1 h). And then we aggregate and transform spatio-temporal data into a 2-D dimensional multi-channel image $X^{(t)} \in \mathbb{R}^{M \times I \times J}$. M, I, J, represents the channel of the image, width, and height, respectively. $X^{(t)}$ is called "spatial traffic flow image", and in this work, M refers to 2, which are inflow and outflow. $S = \left\{s^{0,0}, s^{0,1}, s^{0,2}, \ldots, s^{i,j}\right\}$ is used to represent the attributes of spatial traffic flow image, where $s^{i,j}$ refers to ith rows and jth column area. Fig. 1(c) is the spatial traffic flow image, which is transformed by the traffic volume of regions.

The operation of transforming time-series GPS points information into spatial traffic flow images is necessary. The reasons can be summarized as follows:

- Transforming the GPS points information into traffic flow images based on grid maps can predict the entire city. Compared with road-based traffic flow prediction, prediction areas are wider.
- Transforming into traffic flow images can better extract spatial features (i.e., nearby dependencies and distant correlations). As we all know, CNNs-based methods are very good at extract spatial correlation. Some similar studies [5,30,31] have shown state-of-the-art results in using spatial images to predict traffic flow.
- Transforming into spatial traffic flow images is a key step of data processing for our attention framework. Although other frameworks based on encoder–decoder framework also can process time-series signals and achieves great success, such as Google's neural machine [53] for translation tasks, it depends on a large number of language samples, related word-pieces, semantically, and syntactically rules [46]. And It focuses on context correlations. Different from them, we pay attention to the correlation between different regions in different periods, which needs to convert original data into spatial traffic flow images as input.

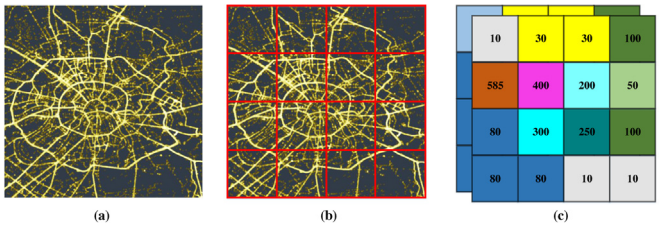


Fig. 1. Data processing on TaxiBJ. (a) is the map of Beijing. (b) is regional division by Definition 1. (c) is the spatial traffic flow image by Definition 2.

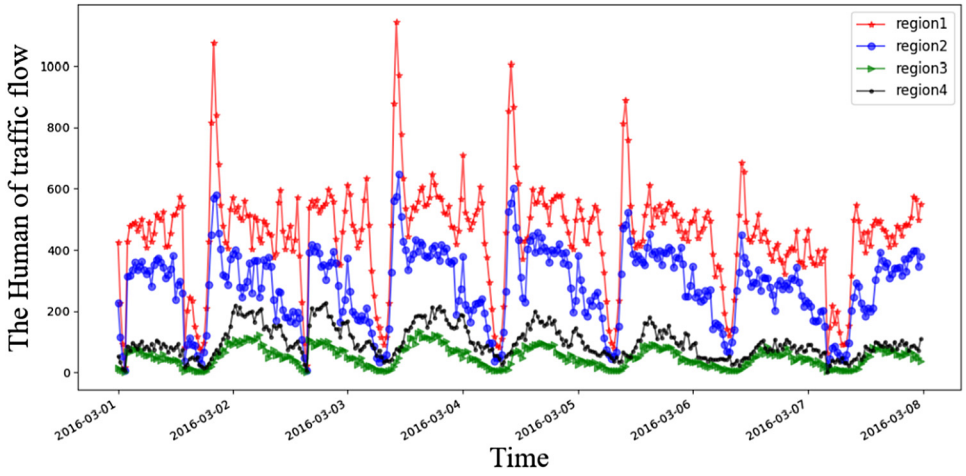


Fig. 2. Four regions of inflow on TaxiBJ data for a week.

Definition 3 (*Inflow and Outflow*)**.** The inflow and outflow of an area $s^{(i,j)}$ at the fixed time interval t are formulated as:

$$x_t^{in,i,j} = \sum_{Tr \in Q} \left| \left\{ k > 1 | g_{k-1} \notin s^{i,j} \wedge g_{k-1} \in s^{i,j} \right\} \right| \tag{1}$$

$$x_t^{out,i,j} = \sum_{Tr \in Q} \left| \left\{ k > 1 | g_{k-1} \notin s^{i,j} \wedge g_{k-1} \in s^{i,j} \right\} \right| \tag{2}$$

where $Tr : g_1 \rightarrow g_2 \rightarrow g_3 \rightarrow \cdots \rightarrow g_{|Tr|}$ is a trajectory in Q, which is acquired at the tth time interval, and g_k is the geospatial coordinate; $g_k \in (i, j)$ means the point of g_k is displayed within the $grid(i, j)$, vice versa, and we calculate by latitude and longitude properly; $|.|$ denotes the cardinality of a set. The traffic flow of Beijing is illustrated in the 3D and 2D images in Fig. 3, which can be expressed the whole inflow and outflow condition. In Fig. 2, we show the inflow pattern of four regions in a week. We can find that the inflow of a region shows a certain periodicity from Fig. 2.

Definition 4 (*Spatial Dependencies*)**.** Spatial dependencies are nearby dependencies and distant correlations.

- **Nearby dependencies**: As shown in Fig. 4, the inflow of region $r1$ is affected by outflow of nearby regions (e.g., $r2$, $r3$, $r4$). Likewise, the outflow of $r4$ would affect inflow from other regions (e.g., $r2$). The inflow of region $r4$ would affect its own outflow as well.
- **Distant correlations**: The flow can be affected by that of farther areas. For instance, people who live in the residential area far from the office area, take the subway or bus from the residential area to the office area, and then the flows in the office area are affected by the residential area (e.g., The flow of $r1$ is affected by $r5$).

Definition 5 (*Temporal Dependencies*)**.** Temporal dependencies include *closeness*, *period*, and *trend*.

- **Closeness**: The flow of human in a region is affected by recent time intervals, both near and far. For instance, traffic congestion occurring at 6:00 a.m. will affect the flow at 8:00 a.m. In this work, if the target is the prediction flow at 10:00 a.m. on Friday, the flow of *closeness* (time interval is 0.5 h) is at 8:30, 9:00, 9:30, and the length of the *closeness* is 3.
- **Period**: Traffic conditions during morning and evening peak interval may be similar for consecutive working days, repeating every 24 h.
- **Trend**: Last Friday's flow of human at 8:00 a.m. may be very similar to this week, repeating every week.

Definition 6 (*Feature-Map Fusion*)**.** Feature-map fusion can also be called late fusion, which is fused in feature maps after a convolution operator. The *closeness*, *period*, and *trend* of feature maps are merged, which can be written as follows:

$$H_{meta}^d = Merge\left(H_C^i, H_P^j, H_S^k\right), \tag{3}$$

where i, j, and k denotes the channels of feature maps of $closeness(H_C)$, $period(H_P)$, and $trend(H_S)$, respectively. H_{meta}^d refers to a mixed multi-channel spatial traffic flow of feature maps, and d is the sum of i, j, and k. After feature-map fusion, the merged features are still connected to the convolution layers, which obtains the information of *closeness*, *period*, and *trend* dependencies.

Definition 7 (*Meta-Data Fusion*)**.** Meta-data fusion is also called early data fusion. The *closeness*, *period*, and *trend* are mainly merged into a multi-channel image by concatenation:

$$X_{meta}^d = Merge\left(X_C^i, X_P^j, X_S^k\right). \tag{4}$$

In Eq. (4), i, j, and k denote the channels of spatial traffic-flow images of *closeness* (X_C), *period* (X_P), and *trend* (X_S), respectively. X_{meta}^d refers to a mixed multi-channel traffic spatial traffic-flow image of meta-data, and d is the sum of i, j, and k.

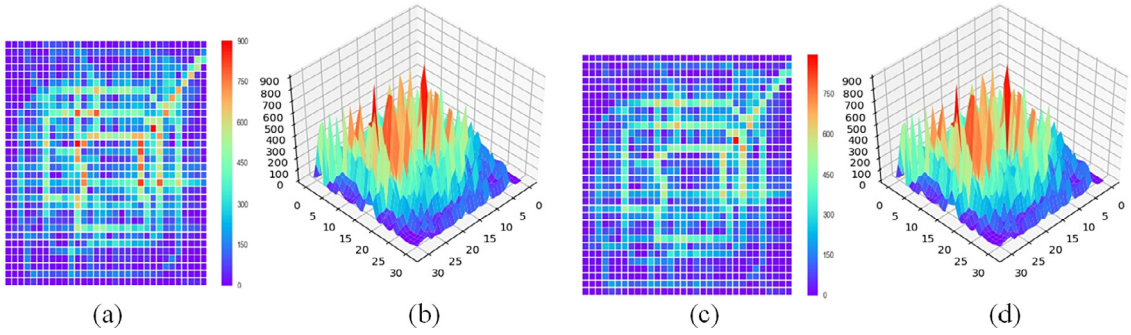


Fig. 3. The 3D and 2D image represent traffic volume in Beijing. (a) is the outflow of the 2D image at 11:00 a.m. in one day, and (b) is the 3D image. (c) is the inflow of the 2D image at 9:00 a.m. on in one day, and (d) is the 3D image at the same time.

Problem 1 (*Single-Step Human Traffic Prediction*)**.** Human traffic prediction aims at estimating the number of people in a given specific area and time interval. Therefore, $X = \{X^1, X^2, X^3, \ldots, X^{n-1}\}$ represents historical observations of a traffic-flow image, and the object is to predict X^n.

Problem 2 (*Multi-Step Human Traffic Prediction*)**.** The main idea of multi-step prediction is that the predicted data are used as input data with which to predict the next period, e.g., two-step forecasting. We obtain X^n by defining Problem 1 and using X^n as the input for the learned model $\mathcal{M}$ to predict X_{n+1}, and then $\{X_n, X_{n+1}\}$ is obtained.

4. Model architecture

In this section, we introduce the detail of ED-ACNN, which consists of three parts, encoder component, the attention blocks of the intermediate unit, and decoder module. Encoder component captures nearby dependencies and distant correlations of grid regions from traffic flow attributes (i.e., *closeness*, *period*, and *trend*). SE block is selected to enhance the channels dependencies of these three traffic attributes' feature maps during the different cases. Decoder module decodes the high-level traffic flow features into the next time-interval traffic flow.

4.1. Encoder component

CNNs are very effective at extracting correlations of spatial features [54], which is one of the reasons that deep-learning-related methods can outperform other approaches. As shown in Fig. 5, the input layer (assume that I is the total number of feature maps), Q_i $(i = 1, \ldots, I)$, is connected to the convolution layer (assume that J is the total number), Q_j $(j = 1, \ldots, J)$, which is based on many local weight matrices ($I \times J$, in total), $w_{i,j}$ $(i = 1, \ldots, I, j = 1, \ldots, J)$. Assuming that the input feature maps are all one-dimensional, each element of a feature map in the convolutional layer can be calculated as

$$q_{j,m} = \sigma\left(\sum_{i=1}^{I}\sum_{n=1}^{F} Q_{i,n+m-1}\omega_{i,j,n} + \omega_{0,j}\right)(j = 1, \ldots, J), \tag{5}$$

where $Q_{i,m}$ is *mth* unit of the *ith* input feature map Q_i, and, similarly, $Q_{j,m}$ is the *mth* unit of the *ith* input feature map Q_j; $w_{i,j,n}$ denotes the weight scalar of the *n*th element; and $w_{i,j}$ refers to connected weight, which connects the *ith* input features maps to the *jth* feature maps. F represents filter size, which determines the receptive field of input feature maps. The pooling layer is used independently for convolution feature maps. The max pooling function is applied for this work, which can be defined as

$$P_{i,m} = \max_{n=1}^{G} \quad q_{i,(m-1)*s+n}, \tag{6}$$

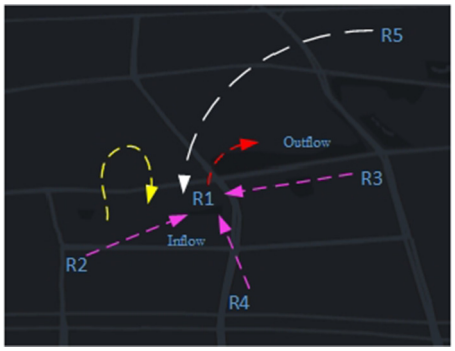


Fig. 4. Dependencies of the nearby and distant.

where G refers to the pooling size; s is the shift size, which is determined by the overlap of adjacent pooling windows.

In the reference process, the advantage of sharing local weight can effectively reduce the number of parameters. Convolution is a filter sliding on the traffic flow image sequence and detecting features in different locations. In this work, the inputs are three sequences of the spatial traffic flow images (i.e., *closeness*, *period*, and *trend*). (1) $\left[X_{t-l_c}, X_{t-(l_c-1)}, \ldots, X_{t-1}\right]$ donates the *closeness* part; (2) $\left[X_{t-l_p.p}, X_{t-(l_p-1).p}, \ldots, X_{t-p}\right]$ is the period part; (3) $\left[X_{t-l_s.s}, X_{t-(l_s-1).s}, \ldots, X_{t-s}\right]$ means the *trend* part. l_c, l_p, and l_s refer to the lengths of *closeness*, *period*, and *trend* sequences, respectively, p and s are a fixed *period* (i.e., a day and a week). With these notations, the convolution over sequences of the spatial traffic flow images can be defined as:

$$H_c^{(1)} = f\left(\sum_{j=1}^{l_c} \omega_{cj}^{(1)} * X_{t-j} + b_c^{(1)}\right) \tag{7}$$

$$H_p^{(1)} = f\left(\sum_{j=1}^{l_p} \omega_{pj}^{(1)} * X_{(t-j).p} + b_p^{(1)}\right) \tag{8}$$

$$H_s^{(1)} = f\left(\sum_{j=1}^{l_s} \omega_{sj}^{(1)} * X_{(t-j).s} + b_s^{(1)}\right) \tag{9}$$

where $*$ represents the convolution operator. W and b denote the learned weight parameters and bias parameters, respectively. H_c, H_p, and H_s refer to the output of the convolution layer in *closeness*, *period*, and *trend* respectively.

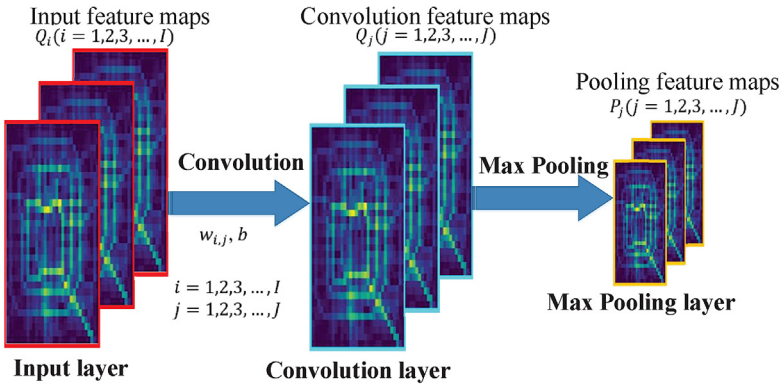


Fig. 5. Convolution ply.

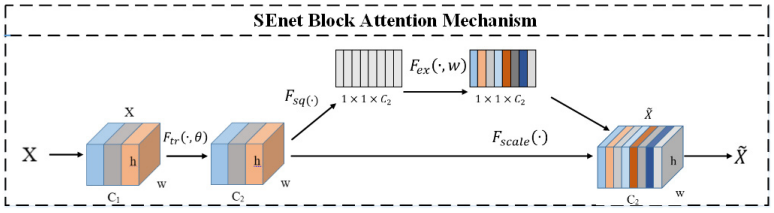


Fig. 6. SEnet block attention mechanism.

4.2. SEnet block for intermediate unit

The Squeeze-and-Excitation (SE) block enhances the representation of the network from the perspective of enhancing the data space dimension. For the multi-channel image after the convolution layer, the SE-block is mainly focused on the channel dimension [52]. The dependencies between the channels are modeled, and the characteristic response values of each channel can be adaptively adjusted. We add SE-block inception to the encoder–decoder framework to extract the spatial-channel weight dependencies features. These dependencies include the extraction of feature maps of *closeness*, *period*, and *trend*, respectively, and the dependencies between them, as well as inflow and outflow. The SE-block includes the squeeze operator (i.e., global information embedding) and the excitation part (i.e., adaptive recalibration).

A squeeze operation squeezes global spatial information into a channel descriptor, which is achieved by using global average pooling to generate channel-wise statistics. Formally, a channel-wise based on statistics $Z \in \mathbb{R}^C$ is generated by global average pooling, and the cth element of Z is calculated by:

$$Z_c = F_{sq}(U_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i,j) \tag{10}$$

where u_c denotes the feature map of the cth channel, $u_{c(i,j)}$ refers to ith row unit of jth column of the feature map of the cth channel; H and W denote the feature map height and width respectively. Z_c is the scalar that is the output of the cth element.

Excitation is primarily about learning nonlinear interactions between channels, and this effect is interdependent. First, it must be capable of learning a nonlinear interaction between channels. Second, it must acquire a non-mutually-exclusive relationship since we would like to ensure that multiple channels are allowed to be emphasized. To meet these standards, the tanh activation function ensures that multiple channels are activated simultaneously.

$$s = F_{ex}(z, W) = \sigma(g(z, W)) = \sigma(W_2 \sigma(W_1 z)) \tag{11}$$

In Eq. (11), σ refers to the tanh function in our approach, $W_1 \in \mathbb{R}^{\frac{C}{r} \times C}$, $W_2 \in \mathbb{R}^{C \times \frac{C}{r}}$ represents weights in two fully connected layers. Z is the scalar that is obtained by the squeeze operator. $\sigma(W_1 z)$ indicates that a fully connected layer is activated by an activation function. In this work, by multiplying the gates of each channel of the corresponding feature maps (i.e., $r = 4, 8, 16$), we can control the flow of information for each feature map. s is the scalar weights. Then, we can obtain the new feature map by multiplying the s and u.

$$\widetilde{X}_c = F_{scale}(U_c, S_c) = S_c \cdot U_c \tag{12}$$

In Eq. (12), S_c is the scalar that represents a set of weights of c channels, U_c can be seen as c feature maps in U, and $\widetilde{X}_c$ refers to the feature maps after the Senet block. It should be noted that the weights S_c and the feature maps $U_c \in \mathbb{R}^{H \times W}$ are matched. Fig. 6 interprets this process of squeezing and excitation.

As shown in Fig. 7, we can obtain features maps of three traffic flow temporal properties after CNN operation. Features maps are fused by concatenating, which can form a multi-channel feature map, written as $U_{c_{merge}}$. We use an SE block to capture the dependencies of feature maps channels of different attributes. This operation can be described as follows:

$$\widetilde{X}_{c_{merge}} = S_{c_{merge}} \cdot U_{c_{merge}} \tag{13}$$

where c_{merge} is the number of feature maps after fused operation and $S_{c_{merge}}$ is the weights that focus on c_{merge} channels. The SE block makes the deep neural networks give different attention to different channels of the feature maps in traffic flow temporal attributes. In different cases, the attention degrees of different channels are different, and the SE block can capture them.

4.3. Decoder module

Deconvolution modules play a role in DL approaches, and is applied in many applications, such as unsupervised feature learning [55,56] and feature visualization of CNNs [57]. [56] introduced a simple framework of overcomplete feature hierarchies for learning parsing, which can capture high-order structure. The

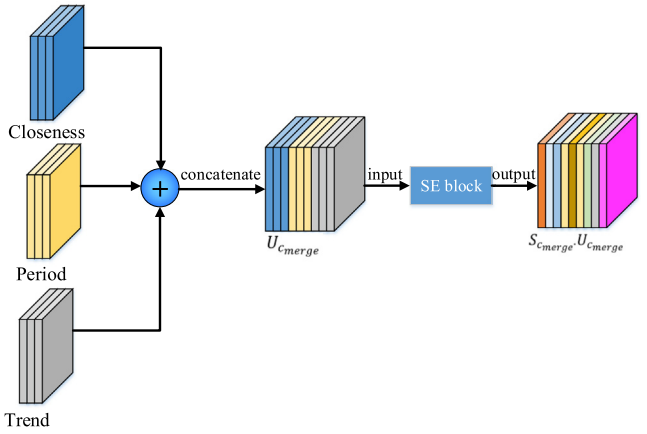


Fig. 7. SE block focuses on feature maps' channels of traffic-flow temporal attributes.

feature map is restored to the pixel space by a deconvolution operator to observe which patterns are more sensitive to the specific feature map [57]. Deconvolution ply as an upsampling operation is used for pixel-wise prediction [48,58]. To obtain full predictions for each pixel, deconvolution modules (more precisely, "transposed convolutions") are used in the present work to up-sample feature maps in the width and height dimensions. Up-sampling of the deconvolution layer can be described as

$$\overline{F} = \sum_{i=1}^{n_1} z_i \otimes f_i^j, \tag{14}$$

where $\overline{F}$ is the reconstructed traffic-flow image, z_i represents *ith* feature map, and f_i^j donates the *ith* deconvolution kernel of the *jth* layer. More precisely, the input feature map is expanded larger by the zero-padding, and one can obtain the up-sampled image by a deconvolution operation. In this work, a deconvolution operation by convtranspose in Keras is used. Fully connected layers are removed as in [48], which is a matter of transforming the feature maps back to another image space and up-sampling. The reasons one can apply a deconvolution module to reconstruct traffic-flow images for the next period are the following.

(1) Many deconvolution kernels with a deconvolution layer contain weight parameters and bias parameters, which can be well trained like CNNs.
(2) Spatial traffic-flow images are used in the present work as the training data, which is closely related to the traffic flow of the next period, and can also be used for pixel-wise prediction in the future.
(3) The previous deconvolution operator is used to restore the original pixel space [48,58], but the goal in this work is to reconstruct the pixel space of a next period of traffic flow, which requires modification of the loss function. The traffic-flow images encoded by CNNs are low-resolution feature maps, and then a deconvolution module is applied to reconstruct the version for the future.
(4) The gating mechanism is used in the present work by forming a bottleneck with convolution layers, max pooling, and deconvolution plies around the learning abilities.

The un-sampled images are controlled by "k", which is the step of the convtranspose operation (i.e., a setting parameter of convtranspose is "stride"). For example, $k = 2$ means that the height and width of the feature map will be magnified 2 times.

4.4. ED-ACNN architecture

ED-ACNN is an adaptively optimized model for learning an approximate function that is based on the encoder–decoder framework. The function is defined as

$$\hat{X}^{(t)} = f\left(\left[X_{meta}^1, X_{meta}^2, X_{meta}^3, \ldots, X_{meta}^{t-1}\right]; \theta\right), \tag{15}$$

where $\left[X_{meta}^1, X_{meta}^2, X_{meta}^3, \ldots, X_{meta}^{t-1}\right]$ is the historical observation of the spatial traffic-flow images, θ denotes the parameters that are needed to learn and represent the future spatial traffic-flow image generated by the approximate function. ED-ACNN is mainly composed of three parts: encoder, attention block, and decoder. An overall structure containing these three parts is shown in Fig. 8. The encoder extracts abstract temporal and spatial features by 2D convolution layers. The intermediate attention structure is an attention mechanism that focuses on the correlation of feature-map channels. The decoder reconstructs the pixel space of the image and generates an image of the next time-interval traffic flow by a deconvolution layer. The traffic-flow image prediction problem is formulated as a regression problem. The proposed method aims to apply the deconvolution module to the regression prediction problems in the field of transportation analysis.

In this work, the encoding process can be written as

$$F_{conv}^i = CNN(T)(i = 1, 2, 3, \ldots, n), \tag{16}$$

where F_{conv}^i refers to *ith* encoding. If $i = 1$, T is the original input data; on the contrary, T is the feature maps. The whole of the intermediate attention structure is described as follows:

$$F_{se} = F_{ex}\left(F_{sq}\left(F_{conv}\right)\right). \tag{17}$$

Feature maps, F_{conv}, are connected to the "squeeze" (i.e., F_{sq}) and "excitation" (i.e., F_{ex}) operations, and the attention feature maps F_{se} are obtained. The decoding process can be summarized as follows:

$$F_{deconv}^j = DeCNN(F_{se})(j = 1, 2, 3, \ldots, n), \tag{18}$$

where F_{conv}^j refers to *jth* decoding. CNNs comprise a top-down approach, and use of a deconvolution module is a bottom-up method. ED-ACNN is a mixed method with both top-down and bottom-up approaches, which attempts to generate next time-interval traffic flow through the deconvolution of mapped features and obtains the same size as the input traffic-flow image. One can use three sequences to predict $\hat{X}$ by minimizing the mean-square error between the predicted and ground truth:

$$\Gamma(\theta) = ||X - \hat{X}||_2^2, \tag{19}$$

where θ is the learned parameters and X is ground truth.

4.5. Training and optimization

Algorithm 1 outlines the training process of ED-ACNN. The training instances are created from the historical sequence data first (*lines*1–4), and then ED-ACNN is trained by back-propagation and Adam (*lines*6–9); the learned ED-ACNN model will then be returned (*line*10).

To better understand the prediction procedure, the flow chart of the proposed method is depicted in Fig. 9. First, GPS-based data is collected, and then these GPS points will be statistically calculated and generate three types spatial flow images by Definitions 1 and 2. Three attributes of traffic flow data are used as input for training model. Next, training samples and prediction samples are prepared. An ED-ACNN deep learning model is established, and its detailed structure is illustrated in Fig. 8. Finally, the testing samples will be fed into the proposed model when the training process is finished, and the testing traffic flow results can be obtained.

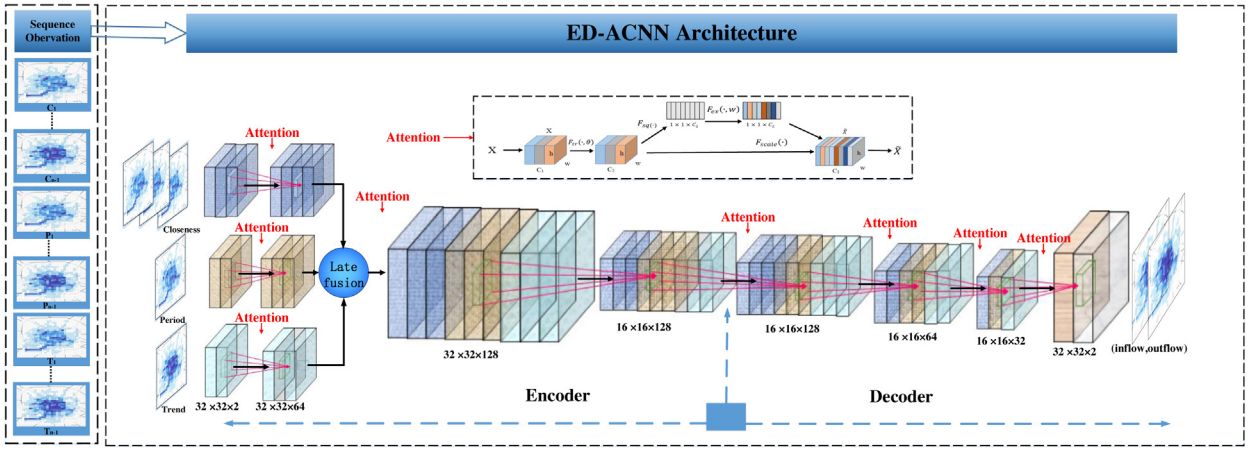


Fig. 8. Overall architecture of ED-ACNN. The network contains convolution (encoder) and deconvolution (decoder) layers, as well as an attention mechanism. Late fusion is used by Definition 6.

Algorithm 1 Framework of ED-ACNN training.

Input:
Given dataset of historical observations;
$S_c = [X_{t-l_c}, \ldots, X_{t-1}]$;
$S_p = [X_{t-l_p.p}, X_{t-(l_p-1).p}, \ldots, X_{t-p}]$;
$S_s = [X_{t-l_s.s}, X_{t-(l_s-1).s}, \ldots, X_{t-s}]$;
LabelData : $Y_{label} \leftarrow X_t$

Output:
ED-ACNN model $\mathcal{M}$;

$\mathcal{D} \leftarrow \varnothing$;
for all available time interval t($1 \leqslant t \leqslant n-1$) **do**
  put $S_c, S_p, S_s, Y_{label} \rightarrow \mathcal{D}$;
end for
Initialize: parameters of the model are θ(includes w_i,b and other parameters);
repeat
  randomly select a batch of instances $\mathcal{D}_b$ from $\mathcal{D}$;
  find θ by minimizing objective formula (19) with $\mathcal{D}$;
until stopping criteria are met (Usually the steps of epochs is reached);
return learned ED-ACNN model $\mathcal{M}$;

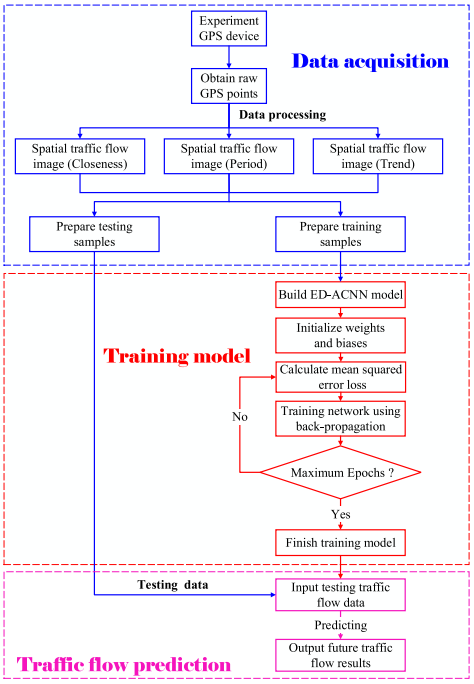


Fig. 9. Flow chart of the proposed method.

5. Experiment

In this section, ED-ACNN and its variants are evaluated on two types of crowd flows in Beijing and New York City and compared with baselines.

5.1. Data and experimental setup

Two different datasets are used: a group of crowd flows from Beijing and one from New York City. **TaxiBJ:** GPS data from more than 34,000 taxis were collected in four parts (1 Jul. 2013–30 Oct. 2013, 1 Mar. 2014–30 Jun. 2014, 1 Mar. 2015–30 Jun. 2015, and 1 Nov. 2015–10 Apr. 2016). The original data are mainly composed of GPS locations, and contain latitude, longitude, time, and taxi passenger status. Using Definition 3, inflow and outflow in every region is obtained. The data of the last four weeks are chosen as test data and the other data for training data. **BikeNYC:** Trajectory data of crowds are obtained from the New York City bike system from April 1 to September 30, 2014. The trip data include trip duration, station IDs, and starting and ending times, which are obtained from more than 6800 bikes. The last 10 d are selected as test data and the rest as training data.

Setting: Table 1 lists the hardware and software on the computer used in the experiments. The popular Keras DL library was adopted.

Hyperparameters: The convolution and deconvolution layers employ 3∗3 filters. 95% of the training data is selected for training each model, and the remaining 5% is chosen as the validation set. Early-stop training is used for pre-training the algorithm based on the best validation score, and the training of the model continues on the full training data for a fixed epoch number. L2 regularization is used to avoid overfitting.

Evaluation index: The proposed method is measured by root-mean-square error (RMSE):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\hat{X}_i - X_i\right)^2}, \tag{20}$$

Table 1
Computer hardware and software used in experiments.

Item	Content
Processor	GPU GTX1060Ti(6g) Ti 4core
Memory	16g
Operating system	Linux
Tensorflow	Tensorflow1.8
keras	2.1.5
Python	Python 3.5

where X_i and $\hat{X}_i$ are ground truth and predicted value, respectively. n represents the number of predictions, $X, \in \mathbb{R}^{M*I*J}$.

5.2. Results

For the evaluation, the proposed method was compared with the following 10 well-known baselines.

HA: Historical average is based on statistics, counting the values at the same time interval, and finally calculating the average.

ARIMA: The autoregressive integrated moving average method is one of the best methods for time-series predictive analysis.

SARIMA: The seasonal autoregressive integrated moving average is a variant of ARIMA and is very popular in time-series prediction.

XGBoost [59]: Extreme gradient boosting is an excellent ensemble learning method that is widely used in many areas.

GRU [46]: Gated recurrent unit as a variant of LSTM is simpler in structure, which is good at dealing with periodic problems.

ConvLSTM [44]: The convolutional LSTM network is designed to primarily process sequence images.

DeepST [29]: A method based on a learning-based prediction model for spatio-temporal data (DeepST).

ST-ResNet [5]: ST-ResNet is a more advanced spatio-temporal model that is based on the unique properties of spatio-temporal data and residual learning.

MST3D [31]: A multiple 3D CNN is an approach to apply 3D CNNs to learn the spatio-temporal correlation features jointly from low- to high-level layers for traffic data.

PCRN [30]: A convolutional recurrent model used to learn traffic patterns and explicit periodic representations or time-series forecasting in geospatial data.

Variants of ED-ACNN: Feature-map fusion is used on the structure of **ED-ACNN** by Definition 6. ED-ACNN includes four encodings and four decodings, as explained in Fig. 8. The structure is simplified and the first decoding layer with the most model parameters is removed, called **ED-ACNN-S**. There are four encodings and three decodings with ED-ACNN-S. Meta-data fusion is mainly used based on the structure of **ED-ACNN-M** by Definition 7. ED-ACNN-M is similar to ED-ACNN, except for feature fusion. ED-ACNN extracts features for *closeness*, *period*, and *trend* separately, and then merges them by Definition 6. Meta-data fusion is used, and then features are extracted from the mixed multi-channel traffic spatial traffic-flow images with ED-ACNN-M.

5.2.1. Single-step prediction

Our experiments were performed on the TaxiBJ and BikeNYC datasets, using historical observations to obtain the single-step prediction of time t. The RMSE of all methods is shown in Table 2, including 10 baselines and the method presented in this work. Our baseline methods include ensemble learning, time series, 3D CNNs, 2D CNNs, and statistical calculation approaches. As can be seen from Table 2, ED-ACNN outperforms all baselines on TaxiBJ. ST-ResNet, MST3D, and PCRN all have good performance, which indicates that DL methods do play a role in traffic-flow prediction. The results of the DL approaches demonstrate that it is practical to use the CNN-based methods to predict crowd flows. The XGBoost algorithm is more robust than the two time-series methods, and the ensemble learning method is better than ARIMA and SARIMA on TaxiBJ. All in all, ED-ACNN achieves the best RMSE. ED-ACNN-S is a simplified version of ED-ACNN, which can be described as an ED-ACNN variant. The error of ED-ACNN-S is higher than that of ED-ACNN because the simplified structural model parameters are too few compared with other DL methods [5,30,31] to fit an excellent model. However, its error is still lower than ST-ResNet, HA, ARIMA, SARIMA, XGBoost, GRU, ConvLSTM, DeepST, as well as ED-ACNN-M. This shows that the proposed method extracts effective and useful features on spatio-temporal data.

Table 2
Performance comparisons of baselines on TaxiBJ.

Method	RMSE	
	TaxiBJ	BikeNYC
HA	57.69	21.58
ARIMA	22.78	10.07
SARIMA	26.88	10.56
XGBoost [59]	17.92	6.98
GRU [46]	22.8	8.71
ConvLSTM [44]	19.63	8.03
DeepST [29]	18.18	7.43
ST-ResNet [5]	16.88	6.33
MST3D [31]	16.05	5.81
PCRN [30]	15.85	–
ED-ACNN-S	16.4	**5.78**
ED-ACNN-M	16.15	5.80
ED-ACNN	**15.7**	6.04

The proposed ED-ACNN-S algorithm achieves the lower RMSE on BikeNYC data compared with other baselines in Table 2. Compared with TaxiBJ, BikeNYC data has fewer grid divisions, which is 16∗8, so an attention SE-block is used in the proposed method. The Beijing area was mapped into $32 * 32$ grids, and there were more grids to forecast. Therefore, seven attention blocks were designed for TaxiBJ, which demonstrates that the more predicted regions may include the more sophisticated feature. Thus, more attention blocks are needed to capture more region information. We try to use ED-ACNN on BikeNYC and find a little over-fitting, so the simplified structure (ED-ACNN-S) is designed. To the best of our knowledge, most of the model parameters can lead to overfitting, so the structure of ED-ACNN-S is more reasonable. Figs. 10(a–x) show the visualization results of the middle layers, including the conv1 layer, conv2 layer, conv3 layer, merge layer, pooling layer, deconv1 layer, deconv2 layer, and deconv3 layer. It can be found that some feature maps [e.g., (a), (b), and (merge6)] have some of the silhouette features of human traffic flow, which shows that some target features have been extracted. Regarding other images [e.g., (merge2), (p), and (q)], what information they capture cannot be determined, and are probably high-level or hidden features.

5.2.2. Attention block analysis

Fig. 11 illustrates the effects of different numbers of SE attention blocks on TaxiBJ. The structure of ED-ACNN contains four convolution and deconvolution layers. Each convolution operation is followed by an attention block, except for the last deconvolution layer. The first three convolution layers are the feature extractions of *closeness*, *period*, and *trend*, and the fourth is the merged features. Fig. 11, taking "encoding attention blocks (3)" as an example, represents the attention based on *closeness*, *period*, and *trend*, and each convolution ply of them has an attention mechanism block. "(3)" denotes the three attention blocks. As can be seen from Fig. 11, seven attention blocks of ED-ACNN achieve the lowest error. The prediction error of the attention block after the convolution layer is higher than the deconvolution

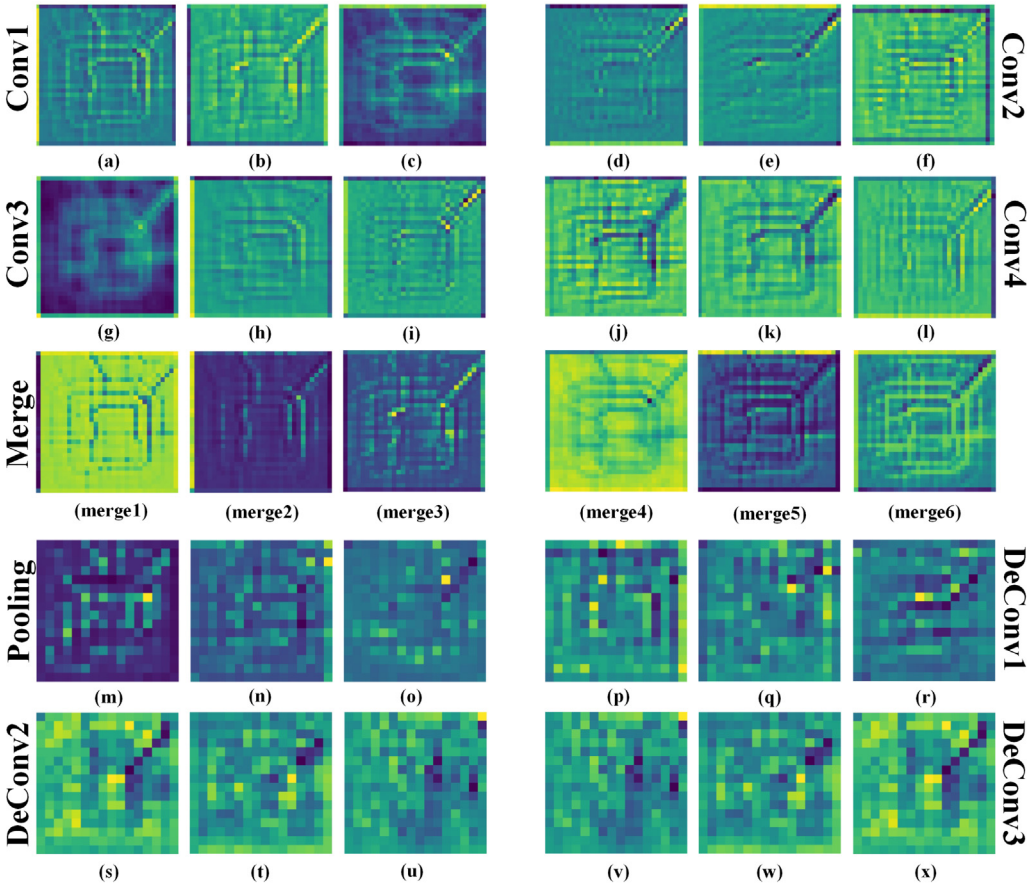


Fig. 10. Feature-map visualization of ED-ACNN on TaxiBJ data.

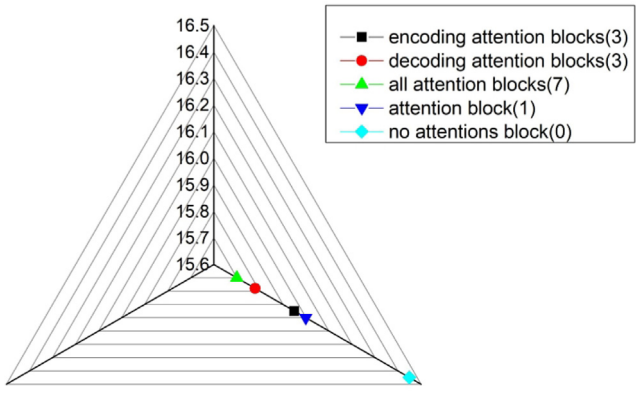


Fig. 11. Effects of different numbers of attention mechanism blocks on TaxiBJ data.

plies, which indicates that the effects of the attention blocks are more vital when reconstructing images. Comparing the RMSE of no attention mechanism block, the results demonstrate that one or more attention blocks are valid.

5.2.3. Volume analysis

Crowd flows can be divided into three categories: high, medium, and low volume. If the area is a $32 * 32$ size, one must predict the traffic of 1024 regions. It can be found that the pattern of the ED-ACNN predicted traffic volume is similar to that of observed traffic flow in Figs. 12(a–f). Our prediction for the three traffic types (i.e., high, medium, and low volume) are stable, and there is no particularly bad situation, which demonstrates the robustness of the proposed model. In fact, one should pay more attention to the situation of high and medium human traffic flow. Hence, using the method proposed in practice is recommended.

5.2.4. Multi-step prediction

Multi-step prediction definition Problem 2 is used. Traffic volume can be predicted over multiple periods, increasing efficiency compared with single-step prediction. As shown in Fig. 13, a four-step prediction experiment was conducted. The multi-step error of ED-ACNN is lower than that of PCRN and ST-ResNet. However, Fig. 13 shows that as the number of prediction steps increases, the error also increases, which indicates that an overly large step prediction is not practical. In actual problems, therefore it is suggested that the number of prediction steps should be less.

5.2.5. Efficiency comparison

The efficiencies of three particularly deep learning models were compared, namely PCRN, ST-ResNet, and MST3D. As can be seen in Fig. 14, the results show that the proposed model has the fewest parameters, smallest model size, and shortest testing time. The proposed method can obtain better performance with fewer parameters, which shows that the proposed method can fit the changes in human traffic volume. ST-ResNet is redundant because of more residual units, model fusion, and fully connected layers. The reason that MST3D has more parameters is due to the fully connected layers. If the prediction regions become abundant, e.g., $64 * 64$, model parameters will exceed the areas of $32 * 32$ and $16 * 8$. However, the proposed method does not rapidly increase the number of model parameters due to the increase of the prediction regions. To the best of our knowledge, training time is generally positively correlated with the

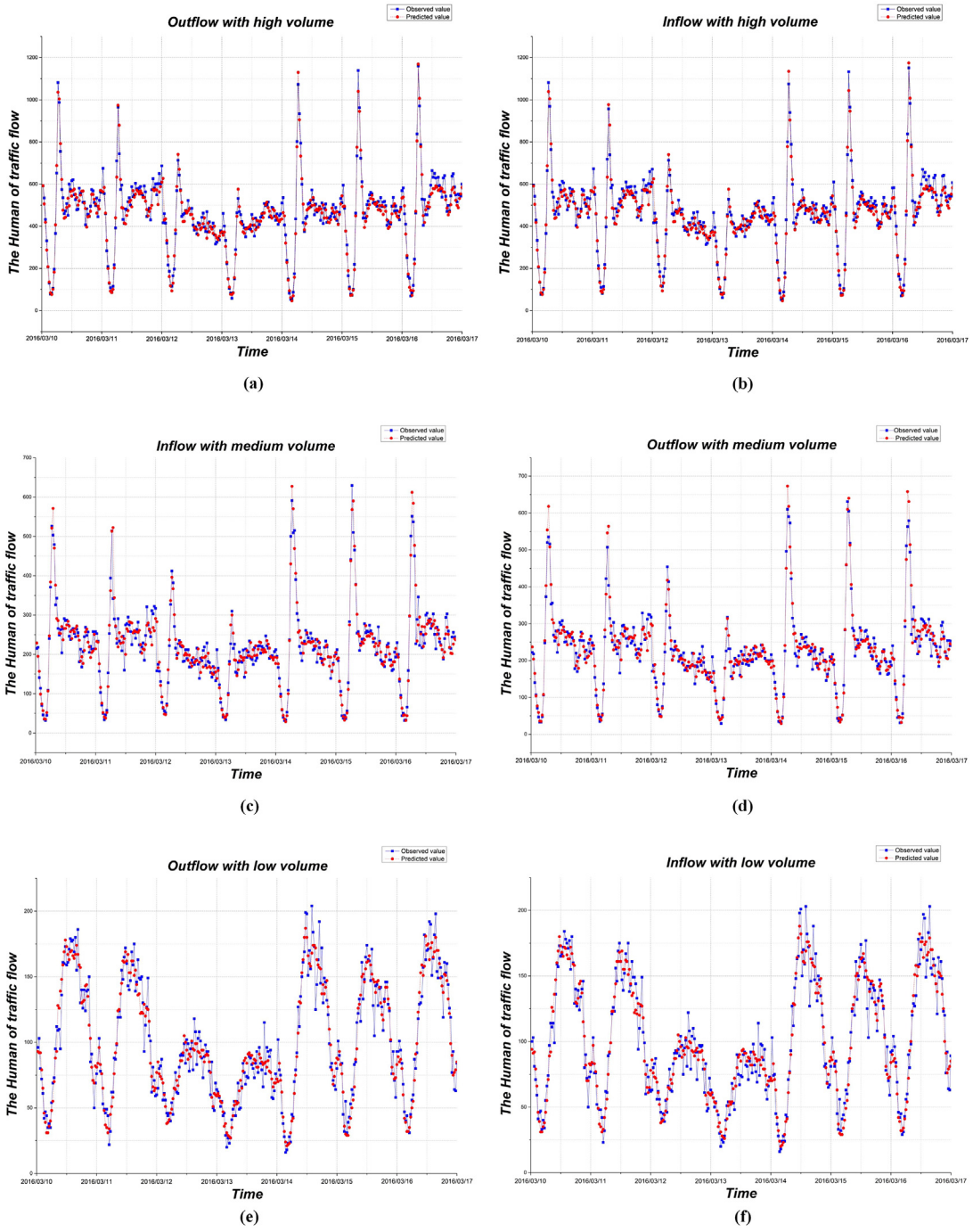


Fig. 12. Predicted inflow and outflow traffic flow with low, medium, and high volume on TaxiBJ data of 1 week.

model parameters, model size, and testing time [60]. As shown in Fig. 14, our simple model (i.e., ED-ACNN-S) features a 2.6 MB model size, 21.36 W parameters, and 0.2 s test time, which is approximately 1/12 that of ST-ResNet. Compared with MST3D and PCRN, ED-ACNN outperforms them in efficiency. As shown in Figs. 14(a–c), ED-ACNN contains 4.6 MB (i.e., 36.6 W parameters) of parameters and the test runtime is 0.37 s. Compared with PCRN, ST-ResNet, and MST3D baselines, the higher instance-level precision is achieved using the proposed model with smaller model size (6.4 times smaller than ST-ResNet, 4.4 times smaller than MST3D, and 1.9 times smaller than PCRN), faster prediction speed (5.8 times faster than ST-ResNet, 3.3 times faster than MST3D, and 43.8 times faster than PCRN). The proposed model also exhibits smaller computation, which proves it can be used in practical problems.

5.2.6. Spatio-temporal property analysis

From the properties of *closeness*, *period*, and *trend*, the *closeness* is more important for traffic-volume prediction. L_C, L_P, and L_S refer to the length of *closeness*, *period*, and *trend*, respectively. Taking L_C as an example, $L_C = 3$ refers to the length of *closeness* being 3. As can be seen from Table 3, only a single data attribute is used for prediction, and the prediction error of the *closeness* is significantly lower than the other two attributes (i.e., *period* and *trend*) in using ST-ResNet and ED-ACNN. This indicates that the property of *closeness* plays a more significant role in the

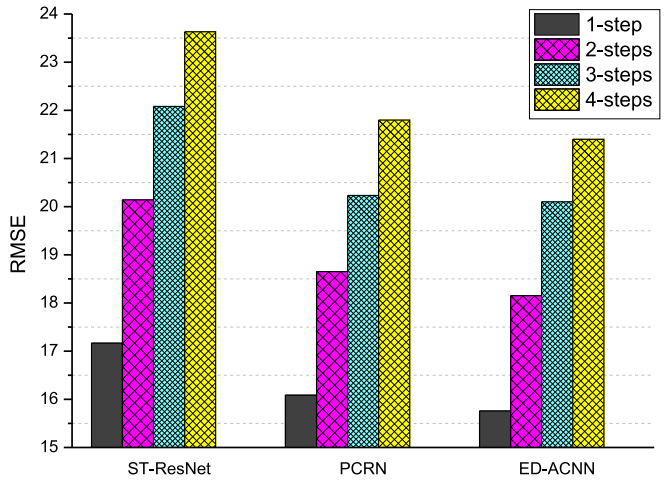


Fig. 13. Four-step prediction on TaxiBJ data.

prediction of traffic flow in DL methods than the other two properties. This condition can be explained as being like a traffic jam at 8:00 a.m. in a region having a great influence on the number of people at 8:30 and 9:00 a.m. The *closeness* feature contains continuous information for the prediction target tendency, e.g., the target is the human traffic prediction at 9:00 a.m. and the *closeness* length is 3, which means that the human traffic flow at 7:30, 8:00, and 8:30 a.m. (assuming that the fixed time interval is 0.5 h). The spatial traffic-flow images of these three periods (i.e., 7:30, 8:00, and 8:30 a.m.) can compose spatial flow information with a 0.5 h interval, which can accurately forecast at 9:00 a.m. Assuming that the target is the human traffic prediction at 9:00 a.m. on Friday, $L_P = 3$ (*period-length* $= 3$) indicates traffic flow at 9:00 a.m. on Tuesday, Wednesday, and Thursday morning. *period* can represent spatial flow information with a time interval of 24 h. Similarly, *trend* can demonstrate the flow of information consisting of spatial traffic-flow images a week apart.

The continuousness of *closeness* (0.5 h), *period* (1 d), and *trend* (1 week) are all related to the prediction target. However, the *closeness* property is most similar to the prediction target tendency. For example, traffic congestion at 8:00 a.m. in a region definitely affects crowd traffic at 8:30 and 9:00 a.m. Traffic congestion at 8:00 a.m. on Tuesday does not demonstrate that congestion will occur at 8:00 a.m. on Wednesday. Similarly, last week's congestion does not mean that there will be congestion this week. These situations are influenced by environmental and external factors. At the same time, the importance of the *closeness* property is reflected in the short-term traffic-flow forecast, especially for 5–15 min forecasting. From the experimental results shown in Table 3, one cannot clearly find that *period* is more important than *trend*; that is to say, for human traffic prediction, 24-h information flow is not necessarily more important than that of 1 week. This may be due to people always resting on weekends and going to work during the week.

The *trend* property of spatio-temporal data is similar to the statistics of HA. As shown in Table 3, a single *trend* attribute is designed for prediction. The errors of ED-ACNN (i.e., 39.32 and 8.91) and ST-ResNet (i.e., 40.33 and 8.93) are much lower than those of HA (i.e., 57.69 and 21.58), which demonstrates that DL methods are better than traditional statistical methods. As illustrated in Table 3, a phenomenon is noted, namely when *period length* $= 1$ the prediction errors of ED-ACNN and ST-ResNet are less than *period length* $= 2$ and *period length* $= 3$ on TaxiBJ. Theoretically, more related attribute data should obtain better prediction results in DL methods, which suggests that there may be some noise or redundancy in our data.

To better understand the attributes of traffic flow, the parameters of temporal *closeness*, *period*, and *trend* before the fusion layer are visualized separately, which facilitates learning different temporal influence degrees for each region of a city, as shown in Fig. 15. Each element based on a grid map in each sub-figure indicates a learned parameter of a certain region that reflects the influence degree by *closeness*, *trend*, or *period*. In Fig. 15(a), the parameters of the three long striped regions are very low, and these regions represent the freeways in the real world. This indicates that these regions have less *closeness*. In contrast, the small blue squares represent the regions near the university town, which has strong *closeness*. As shown in Fig. 15(b), Sun Park and Beijing Zoo are high-*trend* regions, like the Zhong guan cun regions, which have a higher *period* than Xuanwu hospital regions in Fig. 15(c).

6. Conclusion

In this paper, a novel attention convolution neural network based on an encoder–decoder framework (called ED-ACNN) is proposed for forecasting the flow of crowds in every region in an entire city, based on historical human traffic data. The proposed ED-ACNN is capable of learning all spatial (i.e., nearby and distant) and temporal (i.e., *closeness*, *period*, and *trend*) dependencies on traffic-flow images. The performance of ED-ACNN on three types of real-world datasets in Beijing and New York City are evaluated, winning the competitive performance between 10 popular baselines in accuracy and efficiency, which indicates that the proposed method is more applicable to traffic-flow prediction. Meanwhile, it is found that find the *closeness* information is more important compared with the other two properties by analyzing the single attribute of traffic flow. In the future, a human traffic-flow warning decision support system (e.g., Fig. 16) will be deployed on the parallel version to monitor the flow of crowds. In addition, the optimization of ED-ACNN and its application to other datasets (e.g., social media positioning, RFID, and WIFI positioning) comprises our planned future work.

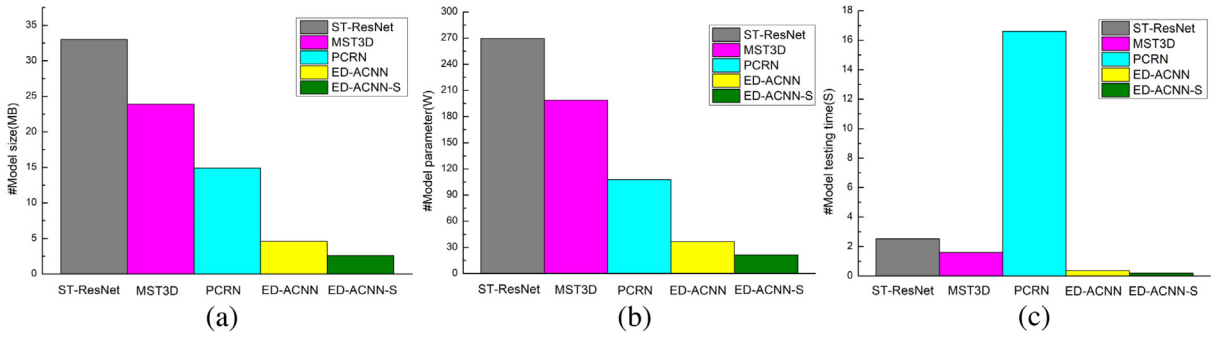


Fig. 14. Comparison with baselines in model size, model parameters, and runtime.

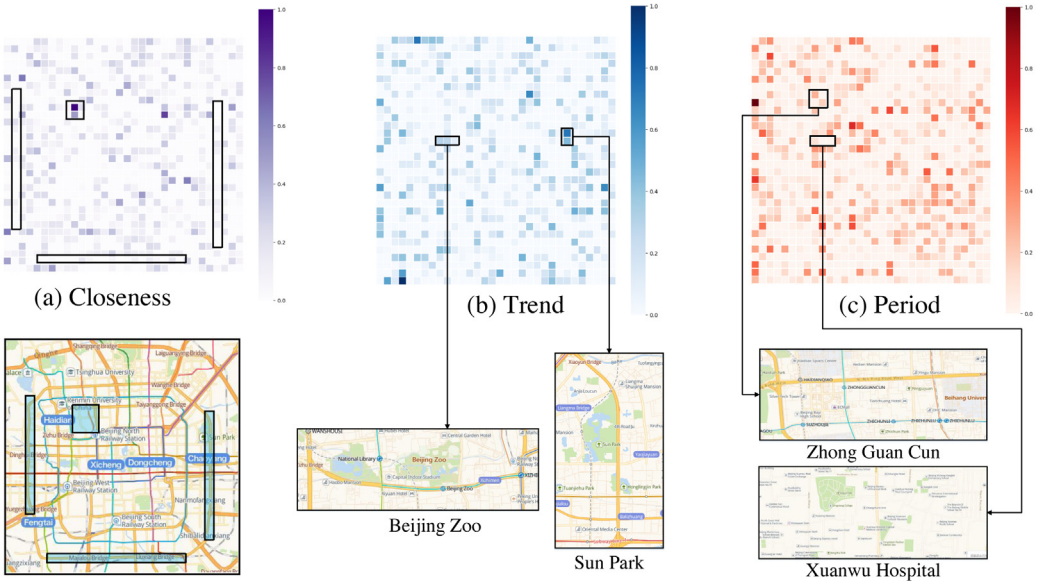


Fig. 15. Visualization of parameters on TaxiBJ.

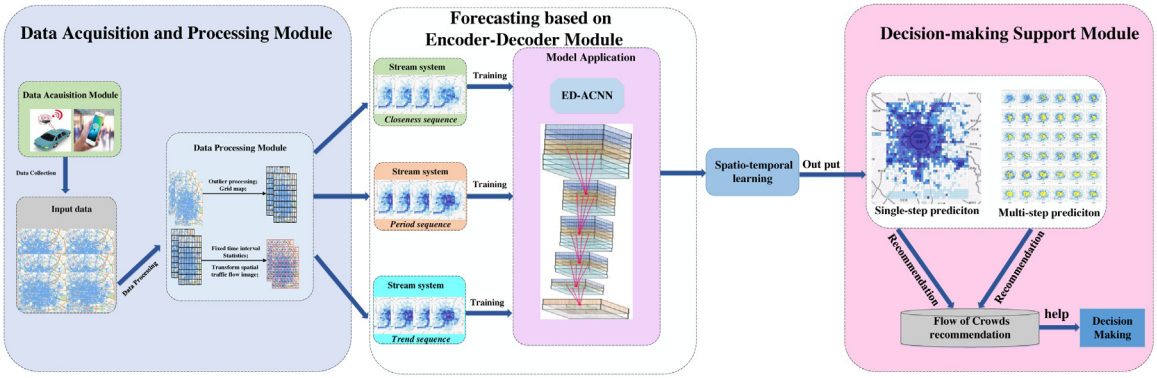


Fig. 16. Framework of human traffic-flow warning decision support system based on ED-ACNN.

Table 3
Comparison with ST-ResNet in using a single attribute on TaxiBJ.

Method	Setting	RMSE	
		TaxiBJ	BikeNYC
ED-ACNN	$L_C = 3$	16.91	6.12
	$L_C = 2$	17.20	6.20
	$L_C = 1$	17.31	6.27
	$L_P = 3$	38.59	9.90
	$L_P = 2$	36.88	10.68
	$L_P = 1$	35.41	11.65
	$L_S = 3$	39.32	8.91
	$L_S = 2$	39.17	8.93
	$L_S = 1$	40.73	9.56
ST-ResNet	$L_C = 3$	19.95	6.38
	$L_C = 2$	21.04	6.46
	$L_C = 1$	21.6	6.51
	$L_P = 3$	39.4	9.97
	$L_P = 2$	37.47	10.37
	$L_P = 1$	36.76	11.41
	$L_S = 3$	40.33	8.93
	$L_S = 2$	37.56	9.15
	$L_S = 1$	38.52	9.43

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported in part by the National Natural Science Foundation of China under Grant 61572177, the National Outstanding Youth Science Program of National Natural Science Foundation of China under Grant 61625202, the Key Program of Natural Science Foundation of China under Grant 61432005 and the International (Regional) Cooperation and Exchange Program of National Natural Science Foundation of China under Grant 61661146006, the Postgraduate Scientific Research Innovation Project of Hunan, China under Grant CX20190309.

References

[1] Mohammad Reza Jabbarpour, Houman Zarrabi, Rashid Hafeez Khokhar, Shahaboddin Shamshirband, Kim-Kwang Raymond Choo, Applications of computational intelligence in vehicle traffic congestion problem: a survey, Soft Comput. 22 (7) (2018) 2299–2320.

[2] Yu Zheng, Licia Capra, Ouri Wolfson, Hai Yang, Urban computing: concepts, methodologies, and applications, ACM Trans. Intell. Syst. Technol. (TIST) 5 (3) (2014) 38.

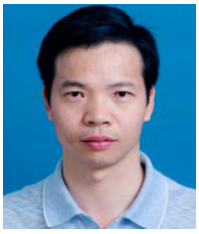
[3] Jin Liu, Xiao Yu, Zheng Xu, Kim-Kwang Raymond Choo, Liang Hong, Xiaohui Cui, A cloud-based taxi trace mining framework for smart city, Softw. - Pract. Exp. 47 (8) (2017) 1081–1094.
[4] Zipei Fan, Xuan Song, Ryosuke Shibasaki, Ryutaro Adachi, CityMomentum: an online approach for crowd behavior prediction at a citywide level, in: Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, ACM, 2015, pp. 559–569.
[5] Junbo Zhang, Yu Zheng, Dekang Qi, Deep spatio-temporal residual networks for wide crowd flows prediction, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017.
[6] Tom Thomas, Wendy Weijermars, Eric Van Berkum, Predictions of urban volumes in single time series, IEEE Trans. Intell. Transp. Syst. 11 (1) (2009) 71–80.
[7] Mohammed S. Ahmed, Allen R. Cook, Analysis of Freeway Traffic Time-Series Data By using Box-Jenkins Techniques, Number 722, 1979.
[8] Mascha Van Der Voort, Mark Dougherty, Susan Watson, Combining Kohonen maps with ARIMA time series models to forecast traffic flow, Transp. Res. C 4 (5) (1996) 307–318.
[9] Billy M. Williams, Lester A. Hoel, Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results, J. Transp. Eng. 129 (6) (2003) 664–672.
[10] Billy M. Williams, Multivariate vehicular traffic flow prediction: evaluation of ARIMAX modeling, Transp. Res. Rec. 1776 (1) (2001) 194–200.
[11] Yiannis Kamarianakis, Poulicos Prastacos, Forecasting traffic flow conditions in an urban network: Comparison of multivariate and univariate approaches, Transp. Res. Rec. 1857 (1) (2003) 74–84.
[12] Jianhua Guo, Wei Huang, Billy M. Williams, Adaptive Kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification, Transp. Res. C 43 (2014) 50–64.
[13] Kranti Kumar, M. Parida, V.K. Katiyar, Short term traffic flow prediction for a non urban highway using artificial neural network, Procedia-Social Behav. Sci. 104 (2013) 755–764.
[14] Chun-Hsin Wu, Jan-Ming Ho, Der-Tsai Lee, Travel-time prediction with support vector regression, IEEE Trans. Intell. Trans. Syst. 5 (4) (2004) 276–281.
[15] Manoel Castro-Neto, Young-Seon Jeong, Myong-Kee Jeong, Lee D. Han, Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions, Expert Syst. Appl. 36 (3) (2009) 6164–6173.
[16] Shiliang Sun, Changshui Zhang, Guoqiang Yu, A Bayesian network approach to traffic flow forecasting, IEEE Trans. Intell. Trans. Syst. 7 (1) (2006) 124–132.
[17] Haiying Li, Yitang Wang, Xinyue Xu, Lingqiao Qin, Hanyu Zhang, Short-term passenger flow prediction under passenger flow control using a dynamic radial basis function network, Appl. Soft Comput. 83 (2019) 105620.
[18] Gary A. Davis, Nancy L. Nihan, Nonparametric regression and short-term freeway traffic forecasting, J. Transp. Eng. 117 (2) (1991) 178–188.
[19] G. Peter Zhang, Time series forecasting using a hybrid ARIMA and neural network model, Neurocomputing 50 (2003) 159–175.
[20] Fabio Moretti, Stefano Pizzuti, Stefano Panzieri, Mauro Annunziato, Urban traffic flow forecasting through statistical and neural network bagging ensemble hybrid modeling, Neurocomputing 167 (2015) 3–7.
[21] Byungkyu Brian Park, Hybrid neuro-fuzzy application in short-term freeway traffic volume forecasting, Transp. Res. Rec. 1802 (1) (2002) 190–196.
[22] Wei-Chiang Hong, Yucheng Dong, Feifeng Zheng, Shih Yung Wei, Hybrid evolutionary algorithms in a SVR traffic flow forecasting model, Appl. Math. Comput. 217 (15) (2011) 6733–6747.
[23] Adam Poole, Apostolos Kotsialos, Swarm intelligence algorithms for macroscopic traffic flow model validation with automatic assignment of fundamental diagrams, Appl. Soft Comput. 38 (2016) 134–150.
[24] Hatem Ben Sta, Quality and the efficiency of data in "smart-cities", Future Gener. Comput. Syst. 74 (2017) 409–416.
[25] Rob Kitchin, The real-time city? Big data and smart urbanism, GeoJournal 79 (1) (2014) 1–14.
[26] Yisheng Lv, Yanjie Duan, Wenwen Kang, Zhengxi Li, Fei-Yue Wang, Traffic flow prediction with big data: a deep learning approach, IEEE Trans. Intell. Transp. Syst. 16 (2) (2014) 865–873.
[27] Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, Fuad E. Alsaadi, A survey of deep neural network architectures and their applications, Neurocomputing 234 (2017) 11–26.
[28] Tigran T. Tchrakian, Biswajit Basu, Margaret O'Mahony, Real-time traffic flow forecasting using spectral analysis, IEEE Trans. Intell. Transp. Syst. 13 (2) (2011) 519–526.
[29] Junbo Zhang, Yu Zheng, Dekang Qi, Ruiyuan Li, Xiuwen Yi, DNN-based prediction model for spatio-temporal data, in: Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, 2016, p. 92.
[30] Ali Zonoozi, Jung-jae Kim, Xiao-Li Li, Gao Cong, Periodic-CRN: A convolutional recurrent model for crowd density prediction with recurring periodic patterns, in: IJCAI, 2018, pp. 3732–3738.
[31] Cen Chen, Kenli Li, Sin G. Teo, Guizi Chen, Xiaofeng Zou, Xulei Yang, Ramaseshan C. Vijay, Jiashi Feng, Zeng Zeng, Exploiting spatio-temporal correlations with multiple 3D convolutional neural networks for citywide vehicle flow prediction, in: 2018 IEEE International Conference on Data Mining, ICDM, IEEE, 2018, pp. 893–898.
[32] Shaojiang Deng, Shuyuan Jia, Jing Chen, Exploring spatial–temporal relations via deep convolutional neural networks for traffic flow prediction with incomplete data, Appl. Soft Comput. 78 (2019) 712–721.
[33] Karen Simonyan, Andrew Zisserman, Very deep convolutional networks for large-scale image recognition, 2014, arXiv preprint arXiv:1409.1556.
[34] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.
[35] Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, Kurt Keutzer, SqueezeNet: AlexNet-level accuracy with 50× fewer parameters and <0.5 MB model size, 2016, arXiv preprint arXiv:1602.07360.
[36] Yu Zhang, William Chan, Navdeep Jaitly, Very deep convolutional networks for end-to-end speech recognition, in: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, IEEE, 2017, pp. 4845–4849.
[37] Ying Zhang, Mohammad Pezeshki, Philémon Brakel, Saizheng Zhang, Cesar Laurent Yoshua Bengio, Aaron Courville, Towards end-to-end speech recognition with deep convolutional neural networks, 2017, arXiv preprint arXiv:1701.02720.
[38] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
[39] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, in: Advances in Neural Information Processing Systems, 2015, pp. 91–99.
[40] Quoc Le, Tomas Mikolov, Distributed representations of sentences and documents, in: International Conference on Machine Learning, 2014, pp. 1188–1196.
[41] Lu Zhao, Yonghua Zhou, Huapu Lu, Hamido Fujita, Parallel computing method of deep belief networks and its application to traffic flow prediction, Knowl.-Based Syst. 163 (2019) 972–987.
[42] Wenhao Huang, Guojie Song, Haikun Hong, Kunqing Xie, Deep architecture for traffic flow prediction: deep belief networks with multitask learning, IEEE Trans. Intell. Transp. Syst. 15 (5) (2014) 2191–2201.
[43] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, Trevor Darrell, Long-term recurrent convolutional networks for visual recognition and description, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 2625–2634.
[44] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, Wang-chun Woo, Convolutional LSTM network: A machine learning approach for precipitation nowcasting, in: Advances in Neural Information Processing Systems, 2015, pp. 802–810.
[45] Nicolas Ballas, Li Yao, Chris Pal, Aaron Courville, Delving deeper into convolutional networks for learning video representations, 2015, arXiv preprint arXiv:1511.06432.
[46] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, Yoshua Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014, arXiv preprint arXiv:1406.1078.
[47] Vijay Badrinarayanan, Alex Kendall, Roberto Cipolla, Segnet: A deep convolutional encoder-decoder architecture for image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 39 (12) (2017) 2481–2495.
[48] Jonathan Long, Evan Shelhamer, Trevor Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.
[49] Olaf Ronneberger, Philipp Fischer, Thomas Brox, U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 234–241.

[50] Yuxuan Liang, Songyu Ke, Junbo Zhang, Xiuwen Yi, Yu Zheng, Geoman: Multi-level attention networks for geo-sensory time prediction, in: IJCAI, 2018, pp. 3428–3434.
[51] Xiang Li, Wei Zhang, Qian Ding, Understanding and improving deep learning-based rolling bearing fault diagnosis with attention mechanism, Signal Process. 161 (2019) 136–154.
[52] Jie Hu, Li Shen, Gang Sun, Squeeze-and-excitation networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7132–7141.
[53] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al., Google's neural machine translation system: Bridging the gap between human and machine translation, 2016, arXiv preprint arXiv:1609.08144.
[54] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al., Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[55] Matthew D. Zeiler, Dilip Krishnan, Graham W. Taylor, Robert Fergus, Deconvolutional networks, in: CVPR, vol. 10, 2010, p. 7.
[56] Matthew D. Zeiler, Graham W. Taylor, Rob Fergus, et al., Adaptive deconvolutional networks for mid and high level feature learning., in: ICCV, vol. 1, 2011, p. 6.
[57] Matthew D. Zeiler, Rob Fergus, Visualizing and understanding convolutional networks, in: European Conference on Computer Vision, Springer, 2014, pp. 818–833.
[58] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, Generative adversarial nets, in: Advances in Neural Information Processing Systems, 2014, pp. 2672–2680.
[59] Tianqi Chen, Carlos Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 785–794.
[60] Kaiming He, Jian Sun, Convolutional neural networks at constrained time cost, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5353–5360.

Bin Pu received the M.Sc. degree in Software Engineering from the National Pilot School of software, Yunnan University, Kunming, China, in 2018. He is currently working toward the Ph.D. degree in Computer Science, Hunan University, China. His research interest includes deep learning on big data and data mining.

Yuan Liu received the M.Sc. degree in computer science from the College of Information Engineering, Xiangtan University, Xiangtan, China, in 2017. He is currently working toward the Ph.D. degree with the College of Information Science and Engineering, Hunan University, Changsha, China. His research topics are many-objective optimization and deep learning.

Ningbo Zhu received a Ph.D. degree in engineering from the computer use and application, Nanjing University of Science and Technology, Nanjing, China, in 2005. He is an Associate Professor at the College of Information Science and Engineering, Hunan University, Changsha, China. His research interests include pattern recognition and intelligent system, digital image processing and network and information security.



Kenli Li received the Ph.D. degree in computer science from Huazhong University of Science and Technology, China, in 2003. His major research areas include parallel computing, highperformance computing, grid and cloud computing. He has published more than 200 research papers in international conferences and journals such as IEEE-TC, IEEE-TPDS, and ICPP. He serves on the editorial board of the IEEE-TC.

Keqin Li is a SUNY Distinguished Professor of computer science. His current research interests include parallel computing and high-performance computing, distributed computing, energy-efficient computing and communication, heterogeneous computing systems, cloud computing, big data computing, CPU–GPU hybrid and cooperative computing, multicore computing, storage and file systems, wireless communication networks, sensor networks, peer-to-peer file sharing systems, mobile computing, service computing, Internet of things and cyberphysical systems. He has published over 690 journal articles, book chapters, and refereed conference papers, and has received several best paper awards. He is currently or has served on the editorial boards of IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Computers, IEEE Transactions on Cloud Computing, IEEE Transactions on Services Computing, IEEE Transactions on Sustainable Computing. He is an IEEE Fellow.