Future Generation Computer Systems 89 (2018) 78–88

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

A novel fuzzy deep-learning approach to traffic flow prediction with uncertain spatial–temporal data features

Weihong Chen [a,b], Jiyao An [a,*], Renfa Li [a], Li Fu [a], Guoqi Xie [a], Md Zakirul Alam Bhuiyan [c], Keqin Li [a,d]

[a] College of Computer Science and Electronic Engineering, Hunan University, China
[b] College of Information and Electronic Engineering, Hunan City University, China
[c] Department of Computer and Information Sciences, Fordham University, USA
[d] Department of Computer Science, State University of New York, New Paltz, NY, 12651, USA

HIGHLIGHTS

- The FDCN approach is proposed to lessen the impact of data uncertainty.
- A model of the fuzzy deep convolution network is designed.
- The tensor is employed to investigate temporal and spatial properties of traffic flow.
- Experiments verify the performance of the FDCN in convergence and accuracy.

ARTICLE INFO

Article history:
Received 30 March 2018
Received in revised form 16 May 2018
Accepted 10 June 2018
Available online 21 June 2018

Keywords:
Deep learning
Fuzzy representation
Residual networks
Traffic flow prediction

ABSTRACT

Predicting traffic flow is one of the fundamental needs to comfortable travel, but this task is challenging in vehicular cyber–physical systems because of ever-increasing uncertain traffic big data. Although deep-learning (DL) methods with outstanding performance recently have become popular, most existing DL models for traffic flow prediction are fully deterministic and shed no light on data uncertainty. In this study, a novel fuzzy deep-learning approach called FDCN is proposed for predicting citywide traffic flow. This approach is built on the fuzzy theory and the deep residual network model. Our key idea is to introduce the fuzzy representation into the DL model to lessen the impact of data uncertainty. A model of fuzzy deep convolutional network is established to improve traffic flow prediction while investigating the spatial and temporal correlation of traffic flow. We further propose pre-training and fine-tuning strategies that efficiently learn parameters of the FDCN. To the best of our knowledge, this is the first time that a fuzzy DL approach has been applied to represent traffic features for traffic flow prediction. Experimental results demonstrate that the proposed approach to traffic flow prediction has superior performance compared with state-of-the-art approaches.

1. Introduction

1.1. Background

In modern society, the number of vehicles has been increasing in cities and on freeways. Many problems related to this increase, such as traffic congestion, may lead to longer time wasted in travel and thus may result in money loss as well as traffic accidents [1,2]. Obtaining accurate and timely traffic flow information is necessary for individual travelers. With the current explosion of traffic flow data, predicting traffic flow using big data is crucial to ensuring safe travel and designing superefficient navigation, which may help travelers make informed travel decisions and improve public safety [3].

Traffic flow prediction on a large scale heavily depends on historical traffic data and other relevant information (i.e., weather conditions and traffic accidents) and is regarded as a key function component in the vehicular cyber–physical system (VCPS) [4]. The VCPS is a complex system with seamless integration of computation, communication and control technology, and advances in the VCPS will enable comfortability, safety and security [5,6]. Deep learning (DL), which is a new method of machine learning, learns useful features by building a multilayer model to achieve accurate

* Corresponding author.
E-mail addresses: whchen@hnu.edu.cn (W. Chen), jt_anbob@hnu.edu.cn (J. An), lirenfa@hnu.edu.cn (R. Li), fu_li@hnu.edu.cn (L. Fu), xgqman@hnu.edu.cn (G. Xie), mbhuiyan3@fordham.edu (M.Z.A. Bhuiyan), lik@newpaltz.edu (K. Li).

https://doi.org/10.1016/j.future.2018.06.021
0167-739X/

image classification or object identification. This method can transcend conceptual learning and has ability to learn more complex knowledge [7]. DL has been applied successfully in prediction tasks, natural language processing, object detection, and motion modeling. As traffic flow prediction is of complexity in nature, deep learning algorithms can be used to represent traffic features without prior knowledge, which exhibits good performance for traffic flow prediction [8].

1.2. Motivations

As a result of traffic data containing significant noise and unpredictable uncertainty, the concept of data ambiguity emerges [9]. Such ambiguity imposes a great challenge on the ability to understand and predict traffic flow. First, the ability to represent input data is limited as variables interact in uncertain ways. Second, the convolutional neural network is not always robust when training data are disturbed by noise. To overcome these disadvantages and improve the accuracy of prediction, fuzzy logic has been introduced for solving many practical problems, including motor control, traffic prediction, and image recognition and classification [10]. Compared with conventional deterministic representations, fuzzy logic representation flexibly constructs rules for reducing the uncertainties in original data and demonstrates competitive performances in both data representation and robustness for dealing with noise.

The membership function and rule base are crucial to the design of the fuzzy logic system. The most straightforward approach is to define membership functions and rules in advance based on research experience and adjust them according to test results. Another way to design the fuzzy logic is to make the system learn from experience. Most of the existing methods are based on the experience of system research to directly define the membership function and then test the output performance. If the test results are not satisfactory, the membership function and rules should be adjusted. This approach heavily depends on human experience and requires human intervention. For pattern classification, the algorithm in [11] first inputs the original data to the hidden layer space in the DL model and then fuzzifies the deep representation at the output layer of the DL model. DL exploits the brain's cognitive mechanism of hierarchical processing of information from layer to layer, which is a breakthrough in machine learning. In practice, task-driven feature learning allows knowledge to be propagated sequentially from the lower layers to the upper layers so that an intelligent way of automatically discovering informative features from data is provided. Another exploration of the fuzzy logic system is to design the membership parameters and fuzzy rules by learning from the actual data [12,13].

Large-scale traffic flow data is complex, which is mainly affected by three complex factors: spatial dependencies, temporal dependencies, and external factors. Given the uncertainty of these factors, traffic flow prediction becomes quite challenging. Thus, various methods have been proposed for traffic flow prediction [12,14]. From a data representation perspective, no-fuzzy and fuzzy methods are used. From a model structure perspective, shallow and deep methods are proposed. Zhang et al. designed an end-to-end structure of ST-ResNet based on the unique properties of spatial–temporal data [15]. This structure employs the framework of the residual networks to model the temporal closeness, period, and trend properties of traffic flow. The proposed method integrates the residual network and convolutional network with traffic flow and external factors to predict traffic flow in each region. However, the fuzzy method can be introduced to DL, which can further reduce the effect of data uncertainty on system performance.

1.3. Our contributions

In this study, we focus on traffic flow prediction of spatial–temporal data with uncertainty to improve the accuracy of prediction by integrating fuzzy logic and deep learning. The FDCN approach is proposed for traffic flow prediction. The proposed approach constructs the FDCN model, which integrates DL with fuzzy representation to alleviate the limitations of shallow methods in traffic flow prediction. Then, the FDCN model is trained in a layered manner to learn general features of traffic flow. In this model, the fuzzy rules are adaptively learned, and the spatial as well as temporal correlations of traffic flow are inherently considered. In addition, the proposed approach demonstrates good performance for traffic flow prediction. The contributions of this study are as follows:

(1) A fuzzy deep-learning approach called FDCN is proposed for traffic flow prediction. The FDCN approach integrates the fuzzy theory with the deep residual network and the fuzzy rules are generated adaptively using the learning algorithm. It explores the fuzzy as well as deep representations to construct features of traffic flow while relatively solving the problem of uncertainty.

(2) A model of the fuzzy deep convolution network is designed for traffic flow prediction, which employs tensor data representation to investigate temporal and spatial properties of traffic flow. The optimized structure of the FDCN is obtained by exploring the number of layers in the model and the regression functions. The FDCN exhibits powerful prediction capability.

(3) The proposed approach is evaluated on the real Beijing taxicab trajectory data set. The results validate the performance of the proposed method compared with state-of-the-art methods.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 presents preliminaries. Section 4 proposes an FDCN model and its learning algorithm. Section 5 discusses experimental results. Section 6 concludes this study and provides direction for future research.

2. Related work

An efficient algorithm to predict traffic flow is crucial for travelers when they are planning their travel. As early as the 1970s, the autoregressive integrated moving average (ARIMA) model was designed for short-term freeway traffic-flow prediction [16]. Since then, many studies have investigated traffic flow prediction. These approaches can be classified into three categories: parametric, nonparametric, and hybrid approaches [17]. The parameter techniques are ARIMA-based models and Kalman-filtering models, which are based on time-series approaches [18]. Kong et al. used one parameter (i.e., velocity) to efficiently predict traffic state, in which information is collected by the global positioning system (GPS) and the curve-fitting and vehicle-tracking mechanism is used [19]. Because of the complexity of the traffic network, however, traffic flow shows a stochastic and nonlinear quality. Time-series approaches tend to be inefficient in predicting traffic flow as well.

Research has paid much attention to nonparametric approaches from the perspective of the traffic network. The widely used nonparametric approaches include k-nearest neighbor (k-NN) methods, the Bayesian network approach, the online learning weighted support vector regression (SVR), and artificial neural networks (ANNs) [20–23]. Chang et al. presented a dynamic multi-interval traffic volume model, which is based on the k-NN non-parametric regression (KNN-NPR) [20]. Kumar et al. proposed a Bayesian network approach for traffic flow forecasting [21]. Jeong et al. presented an online learning weighted support-vector regression method for short-term traffic flow predictions [22]. Kumar et al. applied the artificial neural network (ANN) model for short term

prediction of traffic flow, in which traffic volume, speed, density, time, and day of week are incorporated as input variables [23]. The ANN training algorithm suffers from the problem of local minima, in which one hidden layer is insufficient to describe the complex relations between the inputs and the outputs of the traffic prediction model.

To obtain adaptive models, some studies explore the hybrid approach, which combines several techniques. Su et al. proposed a hybrid traffic flow prediction model, in which the genetic algorithm (GA) was used to optimize the input parameters and the support vector machine was used to update the parameters of the prediction function [24]. Hong et al. presented an SVR model, in which a Genetic algorithm was combined with simulated annealing to obtain the suitable parameters of Gaussian Radial Basis Function (RBF) accurately for traffic flow prediction [25]. Dunne et al. took advantage of an ANN structure with adaptive learning strategies to complete a regime-based traffic state prediction [26]. Min et al. proposed a scalable multivariate spatial–temporal model for predicting the traffic volume and speed jointly [27]. Although the aforementioned hybrid methods are adaptive, it is difficult to say that one method is superior to another method in any situation. One reason for this difficulty is that the proposed methods are developed with a small amount of data, as opposed to big traffic data. Moreover, the accuracy of traffic flow prediction depends on features embedded in the spatial–temporal traffic data and external factors.

Deep learning has attracted research in the domain of traffic state prediction [4,8,12,28]. Huang et al. proposed a deep structure for traffic flow prediction, which consisted of two parts, that is, a deep belief network (DBN) at the bottom and a multitask regression layer on the top [28]. Lv et al. proposed a stacked autoencoder (SAE) model to learn traffic flow features for prediction, in which the logic regression predictor was added on the top layer and the model was trained in a greedy layer-wise manner [8]. Koesdwiady et al. proposed a holistic deep architecture to improve traffic flow prediction, in which DBNs were used for traffic and weather prediction and a correlation between weather parameters and traffic flow was investigated [4].

The uncertainty of traffic data has constantly been the focus of research. To address such a problem, the fuzzy method has been introduced [29–31]. Pongpaibool et al. designed a traffic congestion estimation system using a manually tuned fuzzy logic [29]. Zhang et al. proposed a fuzzy traffic congestion prediction method, which combined the GA with a hierarchical fuzzy system to optimize the rule base for accurate prediction [30]. Shankar et al. presented the fuzzy inference system utilizing traffic density and speed information to evaluate the level of road traffic congestion [31]. Meanwhile, the experimental results showed that the fuzzy system was an effective method for data presentation in traffic state estimation and prediction. However, the membership functions or fuzzy logic rules are chosen subjectively in these methods, and intelligent learning in the fuzzy system for traffic flow prediction remains unsolved. In this study, we present a fuzzy deep-learning approach to solve the challenge of uncertainty in large-scale traffic flow prediction. We aim to achieve accuracy performance of the system by using the fuzzy DL learning method with the optimized structure.

3. Preliminaries

3.1. Fuzzy logic representation

The fuzzy logic representation is the knowledge representation based on fuzzy logic, which processes uncertain information using simple IF-THEN rules. Because of uncertainties among traffic data, fuzzy logic has often been used in traffic flow prediction [32]. Fuzzy logic representation offers an effective solution for learning from uncertain data. Commonly, a fuzzy system includes an input layer, a fuzzy layer, a rules layer, and a defuzzification layer. In the input layer of the fuzzy logic system, the nodes only pass the input value directly to the next layer. In the fuzzy layer, the membership function is performed for a single node, and the output of this node should be the function value. Nodes in the input layer are connected to membership functions, and linguist labels are assigned to each input variable. In the rules layer, the links between nodes are utilized to perform the matching of fuzzy logic rules. The fuzzy logic operations, such as *AND*, are performed on the rule nodes. In the defuzzification layer, the *OR* operation is used to merge the results from the rules. The rules must be entered manually in the traditional fuzzy logic system. Correspondingly, a fuzzy logic system with an adaptive component emerges, in which the rules and the defuzzification process are adjusted adaptively by supervised learning.

3.2. Deep convolutional networks

Deep convolutional networks (DCN) refer to the deep network containing multiple layers of convolutions and the convolutional neural network (CNN) based on classical LeNet is one of the most commonly used DCN [7]. The CNN is built by stacking multiple layers and consists of consecutive convolutional, nonlinear transformation, pooling, and fully connected layers as well as an input and an output, and have led to a series of breakthroughs for prediction tasks. Convolution is performed at the convolutional layer to extract features from the local neighborhood on feature maps in the previous layer. Then the result combined with additive biases is passed on to the next layer through a nonlinear activation function. For the convolutional operation, the value of a unit in the ith feature map in the lth layer, denoted as $o_i^{(l)}$, is calculated as follows:

$$o_i^{(l)} = w_i^{(l)} x^{(l)} + b_i^{(l)},$$

where $w_i^{(l)}$ and $b_i^{(l)}$ are the weight and the bias for the feature map i of the lth layer in the CNN model, respectively, and $x^{(l)}$ is the input of the lth layer in the CNN model. The nonlinear transformation is performed by the activation function, represented by:

$$(y_d)_i^{(l)} = g(o_i^{(l)}),$$

where g is the activation function, i.e., the rectifier linear unit (ReLU) with $g(x) = max(0, x)$.

Deep networks naturally integrated the hierarchical features, and many prediction tasks have also benefited from deep models because of the significance of depth. However, an obstacle to the network with more layers is the vanishing gradient, which hampers convergence from the beginning. A solution to this problem is to add layers as residual mapping in the CNN model, such as residual networks [33]. The existence of this constructed solution indicates that a deeper model should produce no higher training error than its original counterpart. In this study, another structure of the deep convolutional network, namely, a deep residual network, is introduced.

3.3. Problem formulation

Traffic flow prediction is regarded as a critical problem for comfortable driving. We aim to estimate the volume of traffic flow on a road or a region in the future. Predicting traffic flow in every region throughout a city is extremely challenging as it is affected by temporal and spatial dependencies. For temporal dependencies, traffic congestion may occur during morning rush hour on weekdays and may affect traffic at subsequent times. For spatial dependencies, we partition a city into an $I \times J$ grid map

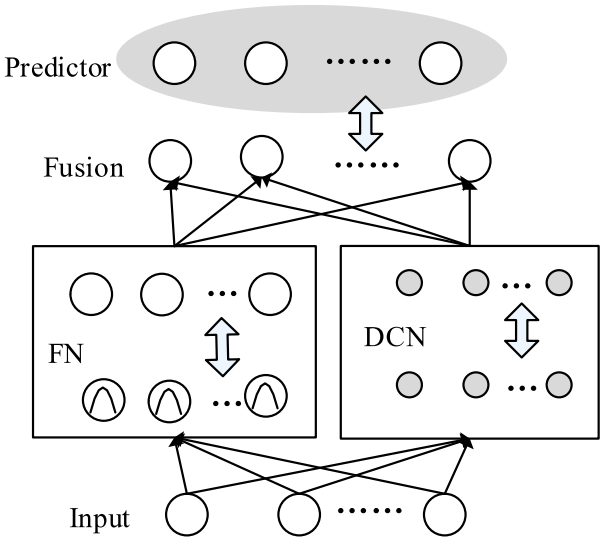

Fig. 1. FDCN model for traffic flow prediction.

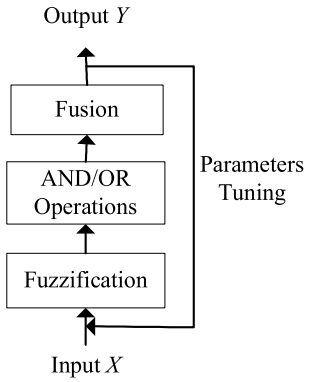

Fig. 2. Fuzzy representation and FN for traffic flow prediction.

in terms of the longitude and latitude, and the inflow of a region is affected by the outflow of its adjacent regions. We denote the attribute values of traffic flow in a city as a tensor $X \in R^{T\times K\times I\times J}$, where T, K, I and J are the time, flow, longitude, and latitude dimensions, respectively. Particularly, traffic flow at time t can be represented by $X_t{=}R^{K\times I\times J}$ for a dynamic system over spatial regions $I \times J$. Thus, this study addresses the traffic flow prediction in a region with spatial–temporal characteristics and external influence factors, and the problem is formulated as follows:

Problem 1. Given a sequence $\{X_t|t = 1, 2, \ldots, n-1\}$ of observed traffic flow data, predict X_n.

4. Fuzzy deep convolutional network and its learning algorithm

In this section, we explain the proposed FDCN methodologies, including the model design and the learning algorithm.

4.1. Fuzzy deep convolutional network model

The proposed FDCN approach for traffic flow prediction is based on the FDCN model, as illustrated in Fig. 1. The FDCN model consists of five modules: the input, the deep convolution network (DCN), the fuzzy network (FN), fusion, and the predictor. Initially, input data flows through the following two channels: one is the FN for fuzzy representation and the other is the DCN for neural representation. After these two modules process the data, the results of each epoch from the FN and the DCN are merged to the fusion module. At this point, the nodes in the fusion module may perform two operation modes: down-up transmission and up-down transmission. The down-up transmission mode should further transmit the fusion result to the predictor, and the up-down transmission mode may calculate the value of the loss function for tuning the parameters in the next step. In the training phase of the FDCN, parameters of the model can be updated repeatedly by minimizing the value of the loss function. Once the model is trained, the predicted result can be obtained by inputting the data into the model.

The structure of the FN is illustrated in Fig. 2. Each node in the input layer is connected with the fuzzification layer, and the membership function is used to calculate the degree that an input node belongs to a certain fuzzy set. It is crucial, however, to determine the membership function in a fuzzy network. The Gaussian membership function is used in the FDCN as follows:

$$u_i = e^{-(x_i-\mu_i)^2/\sigma_i^2}, \forall i, \tag{1}$$

where μ_i is the mean, and σ_i^2 is the variance. In the layer of *AND*/*OR* operations, *AND* is one of the most commonly used fuzzy operations, denoted as follows:

$$(y_f)_i^{(l)} = \prod_{j=1}^{n} u_j = \min\{u_1, u_2 \ldots, u_n\}, \tag{2}$$

where n is the number of nodes on the $(l-1)$th fuzzy layer that connect to node i. The *OR* operations are utilized to perform the matching of fuzzy logic rules by the links between nodes. The output of the fusion layer's nodes is integrated with the results of the DCN part to adjust the fuzzy rules adaptively.

In the DCN part, the deep residual structure is used because the effectiveness of training is subject to the depth of the DCN model. Note that, the fuzzy deep neural network in [12] uses the structure of deep neural network for data classification. For convenience, the model with the fuzzy representation and the CNN structure is denoted as FCNN. In this study, the deep structure with the residual network framework is used for prediction tasks because it has been demonstrated to be quite effective for training a very deep network [33]. The core idea of the residual learning is to learn the additive residual function g with respect to the input of the kth residual unit (ResU) $X^{(k)}$. Formally, a residual unit is defined as follows:

$$X^{(k+1)} = X^{(k)} + g(X^{(k)}), \tag{3}$$

where $X^{(k)}$ and $X^{(k+1)}$ are the input and the output of the kth residual unit, respectively, and g is a residual function. Fig. 3 illustrates the DCN structure with the residual unit that is composed of two convolutions and two nonlinear transformations.

For a spatial–temporal prediction problem, the input may be a long sequence of observation, whose spatial–temporal properties can be challenging to learn. In the DCN part, one convolutional layer can commendably describe near dependency in spatial regions, and a stack of convolutional layers can capture much greater spatial dependency. Thus, the proposed FDCN model explores the temporal and spatial properties of traffic flow for accurate prediction.

4.2. Learning algorithm of the FDCN

The FDCN method explores the fuzzy representation and deep learning to reduce the uncertainty of traffic data. Such multimodal

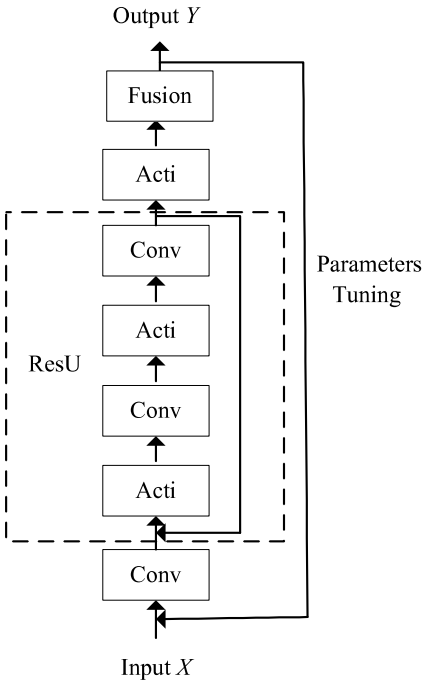

Fig. 3. Deep representation and the DCN for traffic flow prediction.

learning ensures extracting features from various perspectives to capture the complex structure and high-level features of data for prediction [34]. After the whole network structure is established, and the network then enters the learning phase-namely, adjusting the parameters of the membership functions, weights of the connection and biases. In the FDCN model, we combine fuzzy representation and DL to improve the performance of traffic flow prediction. Thus, the fusion layer adopts a densely connected fusion approach, which is calculated by

$$x_i^{(l+1)} = (w_d)_i^{(l+1)}(y_d)^{(l)} + (w_f)_i^{(l+1)}(y_f)^{(l)} + b_i^{(l+1)}, \tag{4}$$

where y_d is the output from the deep representation part, and y_f is the output from the fuzzy representation with weights w_d and w_f, respectively. Then, the fusion result at the fusion layer is deeply transformed by the nonlinear function. The predicted value at the tth time interval, denoted as $\hat{y}_t$, is defined as follows:

$$\hat{y}_t = g(x_i^{(l+1)}) = \frac{1 - e^{-2x_i^{(l+1)}}}{1 + e^{-2x_i^{(l+1)}}}, \tag{5}$$

where $x_i^{(l+1)}$ is the fusion result from the neural representation and the fuzzy representation, and g is the hyperbolic tangent (tanh) activation function. The tanh transformation ensures that the output values are between -1 and 1, that is, close to the normalized input values.

The FDCN model can be trained to predict traffic flow from the input sequence data by minimizing the mean squared error between the predicted value and the true value. The reconstructed error is defined as follows:

$$L(\theta) = \left\| y_t - \hat{y}_t \right\|_2^2, \tag{6}$$

where y_t is the observed value, $\hat{y}_t$ is the predicted value, and θ represents all learnable parameters in the FDCN model.

Before prediction, the FDCN model needs to be trained by parameter initialization and fine-tuning. Better initialization may contribute to the convergence of the neural network to a good local minimum more efficiently. In the FDCN model, parameter initialization includes that of both the FN and the DCN parts. For simplicity, the weights w between all layers in the DCN part are randomly initialized according to the uniform distribution rule:

$$(w_d)_i^{(l)} \sim U\left[-\frac{1}{\sqrt{n^{(l-1)}}}, \frac{1}{\sqrt{n^{(l-1)}}} \right],$$

in which $n^{(l-1)}$ is the number of nodes on the $(l-1)$th layer connected to node i in the lth layer. The bias b for all nodes is initialized as zero. For the fusion layer, the number of nodes is the total number of nodes on the last layer of both the FN and DCN parts. In the FN part of FDCN, parameters that need to be initialized include the weights between layers, as well as the mean value μ_i and the variance σ_i^2 of the membership function. The weights between the "fuzzification" layer and the "AND/OR operations" layer are set to 1. The parameter μ_i is initialized by the statistical method, and σ_i^2 can be determined on the basis of the mean value [35].

When all parts of the FDCN model are well initialized, we can fine-tune parameters in a task-driven manner. To adjust the parameters precisely, the FDCN model is trained by back propagation and the Adam algorithm. The Adam algorithm is suitable for large data sets and high-dimension space as well as most non-convex optimization [36]. The process of updating parameters is described as follows:

(1) Compute the gradient b_t of parameters according to Eq. (6),

$$b_t = \frac{\partial L(\theta)}{\partial \theta^{(l)}} = \sum_n \frac{\partial L(\theta)}{\partial y_i^{(l)}} \cdot \frac{\partial y_i^{(l)}}{\partial x_i^{(l)}} \cdot \frac{\partial x_i^{(l)}}{\partial \theta^{(l)}}, \tag{7}$$

where $L(\theta)$ is the reconstruct error function defined in Eq. (6), θ represents the general parameter set in the FDCN model. The first item in Eq. (7) is the back-propagation term, and the last two terms are derived by the activation function and the output $y_i^{(l)}$ of the neuron.

(2) Update a biased first moment estimate and a biased second raw moment estimate according to Eqs. (8) and (9), respectively:

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) b_t, \tag{8}$$

and

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) b_t^2, \tag{9}$$

where β_1 and β_2 are the exponential decay rates for the first and second moment estimates, respectively; m_{t-1} is the first moment vector, and v_t is the second raw moment vector. Generally, m_0 and v_0 are initialized at zero, and $\beta_1, \beta_2 \in [0, 1)$.

(3) Compute a bias-corrected first moment estimate and a second raw moment estimate according to Eqs. (10) and (11), respectively:

$$\hat{m}_t = m_t / (1 - \beta_1^t), \tag{10}$$

and

$$\hat{v}_t = v_t / (1 - \beta_2^t), \tag{11}$$

where m_t and v_t are the first moment vector and the second raw moment vector in Eqs. (8) and (9), respectively.

(4) Update parameters according to Eq. (12):

$$\theta_t = \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon), \tag{12}$$

where θ_{t-1} represents the value of parameters at time $t-1$, α is the learning rate, and ε is the constant 10^{-8}. In the FDCN model, the parameter set is $\theta = \{W, b, \mu, \sigma\}$. According to the suggestions in [37], we empirically set the learning rate to 0.0002.

After parameter initialization, the model is pre-trained by fine-tuning and a learnable model can be obtained for traffic flow prediction. The training algorithm of the FDCN model is summarized as Algorithm 1. The core details are explained as follows:

(1) In Line 1, the structure of the FDCN model is constructed, which contains the DCN module and the FN module.

(2) In Line 2, the parameters in the FDCN model are initialized as mentioned in Section 4.

(3) In Lines 3–9, the FDCN model is trained until the stopping criteria is met. The training process includes two phase: forward learning and backward propagation.

(4) In Line 5, forward learning is performed to get a predicted value. In the forward learning phase, the preprocessed data is firstly input into the model, and follows the FN module for fuzzy representation and the DCN module for deep learning, respectively. Then, the results from these two modules are fused using Eq. (4). Finally, the fusion result is transformed using Eq. (5) and the predicted value is obtained.

(5) In Lines 6–8, backward propagation is done and the parameters are fine-tuned using the Adam algorithm. In the backward propagation phase, the reconstructed error $L(\theta)$ is computed using Eqs. (6), and the parameters are updated according to Eqs. (7)–(12). The objective of fine-tuning the FDCN model is to minimize the error in Eq. (6).

(6) In Line 9, the training process ends when the stopping criteria is met. The stopping criteria includes the early stopping strategy and the maximum number of epochs. Once the training is completed, a well-trained model is obtained.

Algorithm 1 Training of the FDCN

Input: Given training samples and their labels $D = \{X_0, ..., X_{n-1}\}$.
Output: The FDCN model (W, b, μ, σ)
1: Construct the FDCN model shown in Fig. 1;
2: Initialize the parameters $\theta = \{W, b, \mu, \sigma\}$ and the learning rate α, as mentioned in Section 4;
3: **repeat**
4: Randomly select a batch of instances D_b from D;
5: Forward learn training samples through the FDCN using Eqs. (1)-(5);
6: Compute the error $L(\theta)$ using Eq. (6);
7: Back-propagate the error through the FDCN and update the parameters according to Eqs. (7)-(12);
8: Find θ by minimizing the objective (6) with D_b;
9: **until** Stopping criteria is met;
10: **Return** $M(W, b, \mu, \sigma)$.

5. Experimental verification

This section presents performance comparisons of the FDCN algorithm with ARIMA [17], DeepST [15], CNN [37], FCNN, and FDCN in the similar application scenarios, in which the proposed approach demonstrates effectiveness and merit in comparison with existing approaches.

5.1. Settings

We implemented our method in Python with Keras and Tensorflow libraries. The experiments are conducted on a 64-b Ubuntu platform with an Intel Core i9 3.3 GHz CPU, NVIDIA TITAN XP 12G and 48 GB memory on the TaxiBJ data set. A GPU-accelerated library for deep neural network cuDNN is used, which provides highly tuned implementations for forward and backward convolution, pooling, normalization and activation layers.

The TaxiBJ data set is trajectory data from Beijing taxicab GPS, including four time intervals: 1 July 2013 to 30 October 2013, 1 March 2014 to 30 June 2014, 1 March 2015 to 30 June 2015, 1 November 2015 to 10 April 2016. We obtained multi-attribute data sampling every 30 min on a grid map and obtained 48 samples per day. The features of these data are expressed as input flow, output flow, time information and external factors (holiday, weather, temperature, accident, etc.). After obtaining the original data, incomplete data needs to be removed. We considered data per day whose sample size was less than 48 to be incomplete. To scale the data into the range of -1 and 1, we normalized the

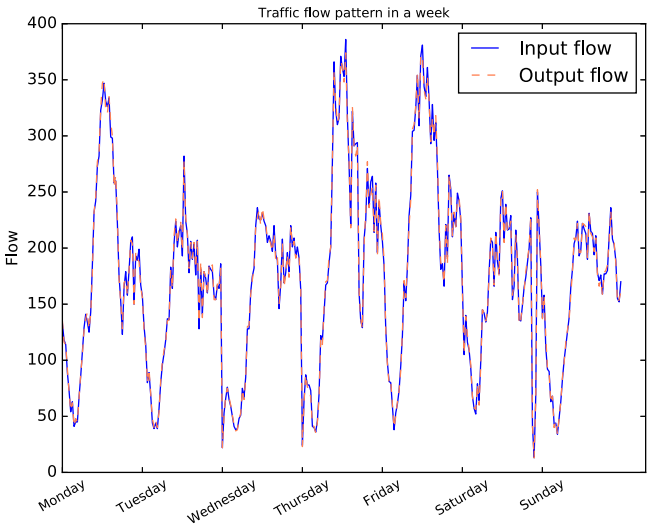

Fig. 4. Typical traffic flow pattern in a week.

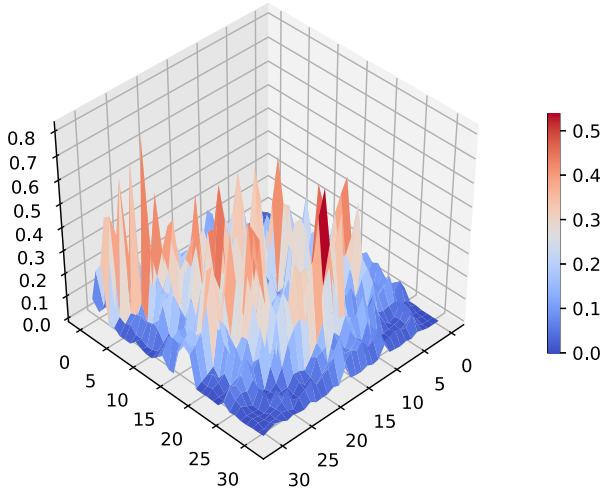

Fig. 5. The input traffic flow of the entire Beijing city at 8 a.m. in one day.

data. The split of the testing data set and the training data set is done using the cross-entropy validation method. In the evaluation phase, the predicted value is re-scaled back to the normal values. Assume that the historical data $X^{t-1}, X^{t-2} \ldots, X^{t-r}$ on 32×32 grid regions with the in and out flows are given. For the prediction tasks of traffic flow, we have three dimensional data with a size of $2 \times 32 \times 32$ as the input; the output is similar. Fig. 4 depicts typical input and output traffic flow patterns of a region over time in a week. As illustrated in Fig. 4, it is observed that traffic congestion may occur at eight in the morning and at noon. Figs. 5–8 show the input and output traffic flows of the entire city of Beijing at these two times. In the Figures, the X axis and the Y axis represent the row and the column of an $I \times J$ grid map, respectively. The Z axis represents the traffic flow that has been normalized. The larger the value of the Z axis is, the greater the traffic flow is. From these four figures, we can see that high traffic flow appears in the center of the city, whereas the low traffic flow is on the edge of the city. In addition, traffic congestion is more prone to happen at noon than at eight in the morning.

To confirm the effectiveness of our models, the experiments are conducted to compare the proposed method with four baselines as follows. The ARIMA method is a general prediction method based

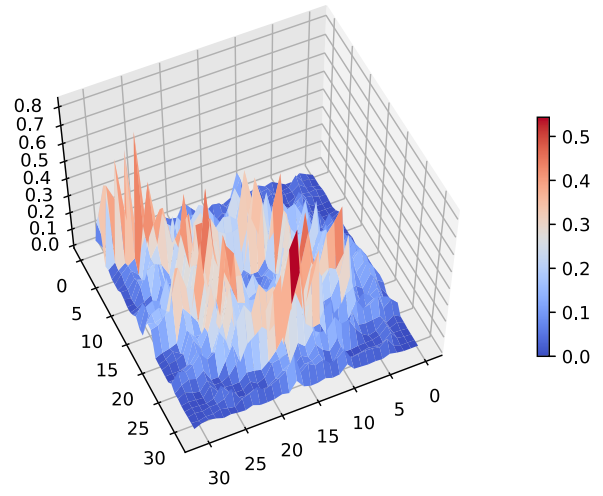

Fig. 6. The output traffic flow of the entire Beijing city at 8 a.m. in one day.

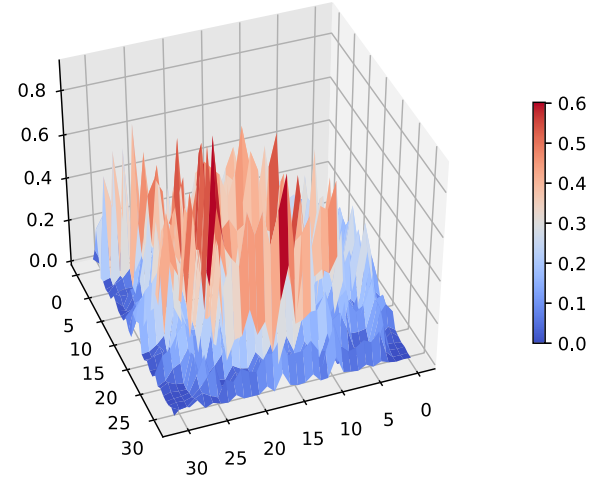

Fig. 7. The input traffic flow of the entire Beijing city at 12 a.m. in one day.

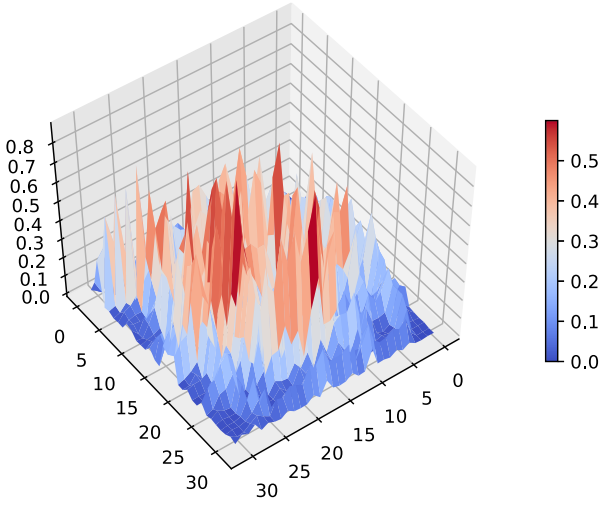

Fig. 8. The output traffic flow of the entire Beijing city at 12 a.m. in one day.

on time series data. The in-flow/out-flow of taxis in our scenarios is a time series, thus it can be predicted by the ARIMA. The DeepST method is a DNN-based prediction model for spatial–temporal data, which combines the closeness feature with the period and the trend features to predict crowd flows with state-of-the-art results. The CNN method is a prediction method using the CNN model, which stacks multiple layers of the convolutional layer and the activation layer. The FCNN method is a prediction method based on the Gaussian fuzzy theory and CNN. The FDCN method is our proposed method in this study, in which the fuzzy representation and the residual network model are used.

5.2. Evaluation metric

To evaluate the effectiveness of the proposed model, we use three performance indexes: the mean absolute error (MAE), the mean relative error (MRE), and the root mean square error (RMSE). These indexes are defined as follows:

$$MAE = \frac{1}{m}\sum_{i=1}^{m}\left|y_i - \hat{y}_i\right|,$$

$$MRE = \frac{1}{m}\sum_{i=1}^{m}\frac{\left|y_i - \hat{y}_i\right|}{y_i},$$

$$RMSE = [\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2]^{\frac{1}{2}},$$

where y_i and $\hat{y}_i$ are the observed value and the predicted value, respectively, and m is the number of all predicted values.

5.3. Experimental results

The experimental results are mainly discussed from three perspectives. First, we verified the convergence of the FCNN and the FDCN model on the TaxiBJ data set. Then, we discussed the optimized structure of the FDCN model by analyzing effects of design factors. Furthermore, we compared the performance of the proposed FDCN model with the ARIMA method, the DeepST method, the CNN method and the FCNN method.

5.3.1. Convergence analysis

To gain insight into the convergence of FDCN, we provide details of the learning process. In the experiments, we performed the parameter initialization as explained in Section 4. The prediction task is set to one-step prediction for FDCN and FCNN models with 16 convolution layers. The mini-batch learning scheme with 48 samples in each batch is employed to speed up the training processes. One epoch means one learning process over all training samples, and all parameters are updated one time after each mini-batch is finished. The early stopping strategy is used to train the FDCN for 30 epochs. The curve of the RMSE with respect to the number of epochs is provided in Fig. 9. From the curve in Fig. 9, it can be observed that the curve flattens as the number of training epochs increases. This indicates the convergence of FDCN and FCNN. In addition, the FDCN model produces less RMSE than the FCNN model, that is, the FDCN model can learn the representation of the TaxiBJ data better than the FCNN model. The whole FDCN model is non-convex, and the converged point may be one of the expected local optima.

5.3.2. Effects of design factors

In the design of the optimized structure of the FDCN model, the considered factors included the number of layers (i.e., the number of convolutional layers) and the regression functions.

The number of layers in the FDCN model depends on the number of stacked residual units, which is a significant design factor in this study. Generally, the accuracy rate can be improved by more layers in the prediction model as the computation time increases.

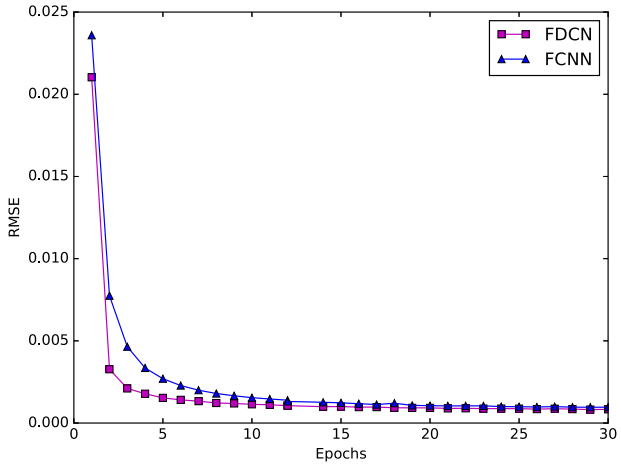

Fig. 9. Learning processes of the FDCN and the FCNN on the TaxiBJ data set.

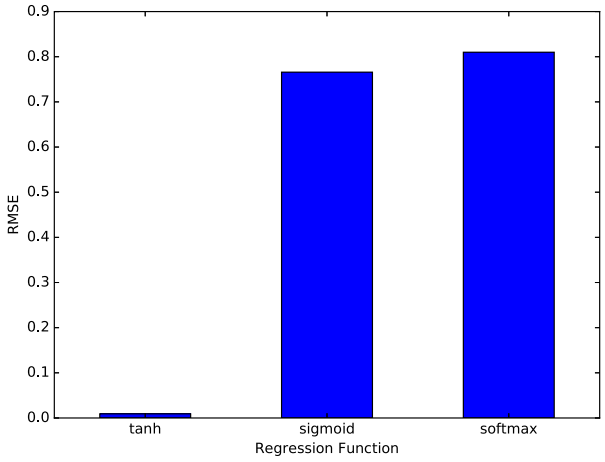

Fig. 10. The RMSE of the FDCN with different regression functions.

It is crucial to find the tradeoff between the performance and the computing cost. This experiment investigated the performance of the FDCN with a varying number of layers for traffic flow prediction. In addition, we compared the proposed FDCN method with the CNN model, the FCNN model, and the DeepST model. Among these four competition models, each had a deep structure and could be applied to the traffic flow prediction scenarios. In all cases, we used the same data set. At the top of the model, we used the tanh function for regression.

In Table 1, we can find that the RMSE of the FDCN decreased as the increase of the number of model layers increased. In the same scale of layers, the FDCN had the lower RMSE than the DeepST and the CNN. In addition, the RMSE of the 20-layer FDCN was 0.3037, which was 98% less than the RMSE of the 20-layer and 24-layer DeepST. When the number of layers in the model was no more than 10, the FCNN obtained the lower values of RMSE and MRE than the FDCN. Nevertheless, the performance of the FDCN was significantly better than the FCNN when the number of layers was greater than 10. When the number of layers increased to 20, the FDCN model obtained the best performance with an RMSE of 0.3037 and an MRE of 0.2045. The average accuracy (1-MRE) of 79.55% using the 20-layer FDCN was roughly equivalent to that of the DeepST model with a layer number of 24; this indicated that the FDCN model could obtain good performance with fewer layers, which simplified the model structure. Such results indicated that the FDCN was more efficient than state-of-the-art deep models.

With regard to the regression functions at the top of the FDCN model, the tanh function has been used commonly in recent years, along with the conventional sigmoid function and the softmax function [7]. This experiment is conducted to compare the root mean square error of the FDCN approach using different regression functions. We set the regression function of the FDCN model as the tanh function, the sigmoid function and the softmax function, respectively. As illustrated in Fig. 10, the FDCN model with the tanh function obtained the lowest RMSE among these three regression functions. Such results indicated that the FDCN using the tanh regression function prompted the predicted value to be closer to the observation.

5.3.3. Multi-step ahead prediction

This experiment is conducted to compare the predicted values of traffic flow using the proposed FDCN method to the observed values for different levels of traffic flow load. The results are illustrated in Figs. 11–13, where the observed traffic flow is also included for comparison. In Figs. 11–13, we observed that the pattern of the predicted traffic flow was similar to that of the observed traffic flow. In addition, the proposed method performed well in high and medium traffic flow conditions. However, this was

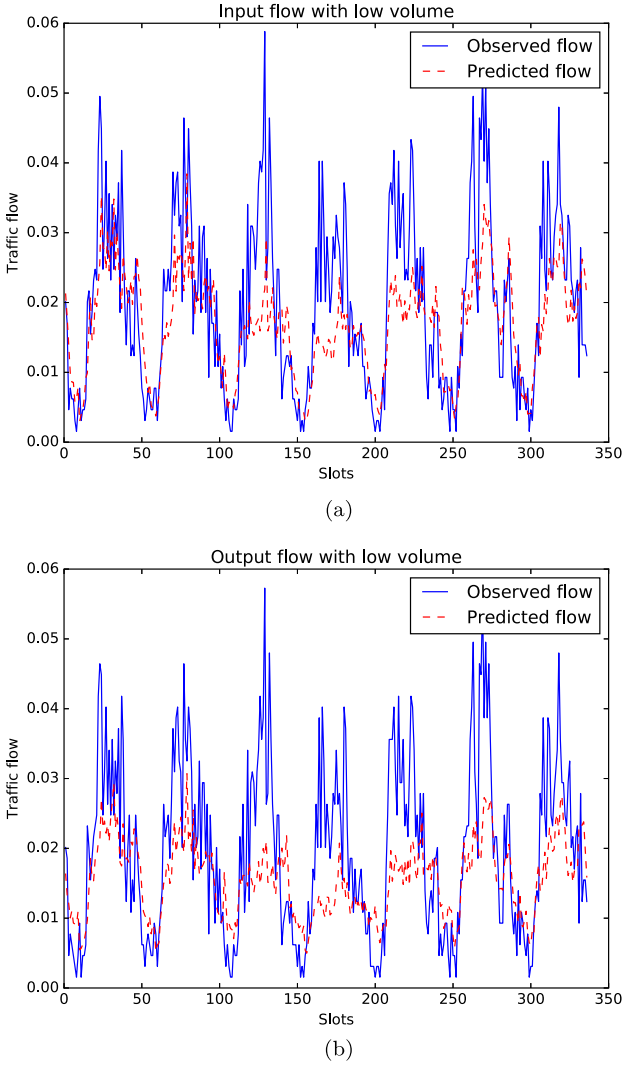

Fig. 11. Predicted input and output traffic flows with low traffic volumes.

Table 1
Performance comparisons of the four model with variant layers.

Layers	DeepST			CNN			FCNN			FDCN		
	RMSE	MRE	MAE	RMSE	MRE	MAE	RMSE	MRE	MAE	RMSE	MRE	MAE
2	23.3604	0.2313	0.0230	23.3545	0.2322	0.0230	0.4245	0.2318	0.0229	2.2168	0.3603	0.0224
4	22.3530	0.2265	0.0186	20.5218	0.2088	0.0187	0.3867	0.2226	0.0211	2.1143	0.3688	0.0502
6	21.9253	0.2173	0.0183	19.9466	0.2146	0.0182	0.3336	0.2184	0.0192	1.7866	0.3726	0.0466
8	21.1351	0.2165	0.0202	21.5762	0.2341	0.0204	0.4319	0.2615	0.0227	1.1898	0.3266	0.0369
10	20.2035	0.2038	0.0187	138.9365	0.8799	0.1430	0.3476	0.2253	0.0197	1.6327	0.3810	0.0414
12	19.8306	0.2115	0.0182	147.0983	0.9745	0.1429	16.7581	1	0.1430	0.6996	0.3188	0.0305
14	21.6321	0.2363	0.0205	147.1442	1	0.1430	16.7581	1	0.1430	0.3336	0.2288	0.0203
16	19.0306	0.1944	0.0173	147.1442	1	0.1430	16.7581	1	0.1430	0.4336	0.2498	0.0231
18	20.5252	0.2067	0.1780	147.1061	1	0.1430	16.7581	1	0.1430	0.3195	0.2188	0.0188
20	19.8368	0.2110	0.0181	147.1442	1	0.1430	16.7581	1	0.1430	0.3037	0.2045	0.0182
22	19.8109	0.2125	0.0182	147.1442	1	0.1430	16.7581	1	0.1430	16.7576	0.9993	0.1430
24	19.4921	0.2045	0.0177	147.1442	1	0.1430	16.7581	1	0.1430	0.3039	0.2206	0.0178

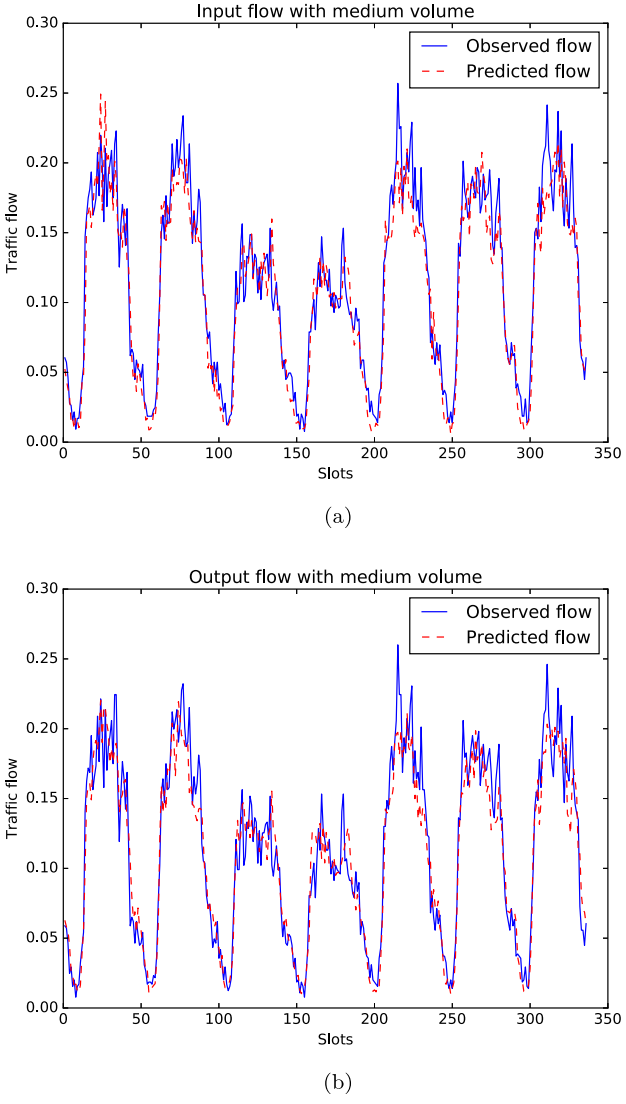

Fig. 12. Predicted input and output traffic flows with medium traffic volumes.

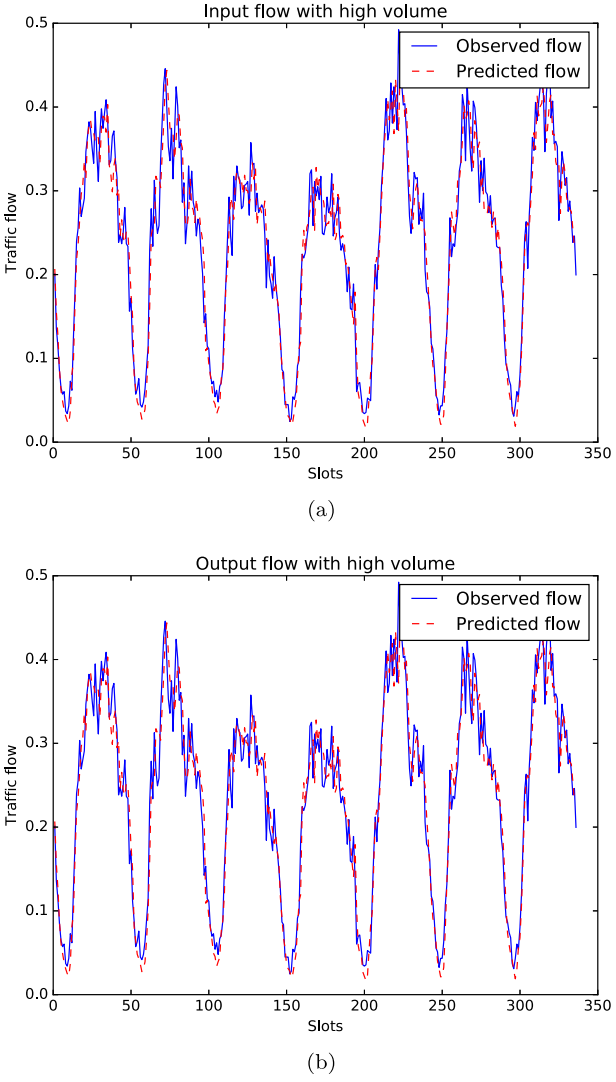

Fig. 13. Predicted input and output traffic flows with high traffic volumes.

not the case in low traffic flow conditions. The main reason for this was that a large relative error may have occurred when the traffic flow rate was small. In fact, we paid more attention to the condition of medium and high traffic flow. Hence, the proposed method effectively predicted traffic flow in practice.

To further verify the performance, the experiment is conducted to compare the RMSE of five methods (i.e., the ARIMA, the CNN, the FCNN, the DeepST and the FDCN) for a varying number of prediction steps. In this study, we use the proposed method to predict 30 min traffic flow, 60 min traffic flow, and so on. In the multi-step forward prediction, historical and recently predicted data could be used to predict traffic flow in subsequent time intervals. The number of prediction steps was set from 1 to 12, and the interval of each step was 30 min. The optimized structure of the models was

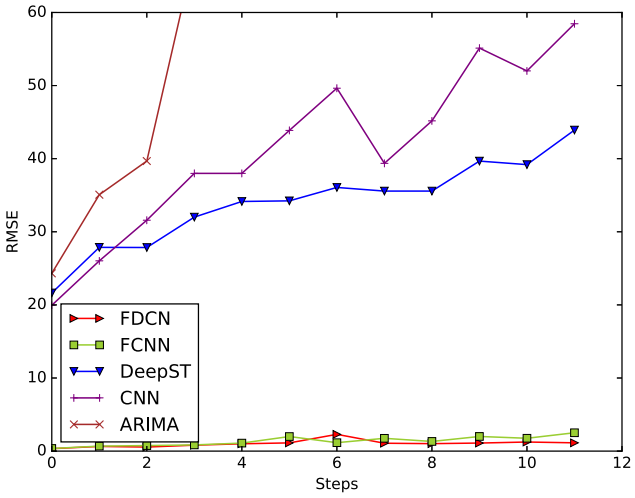

Fig. 14. The RMSE for multi-step ahead prediction.

employed. According to Table 1, we selected the 2-layer CNN, the 2-layer FCNN, the 14-layer FDCN and the 14-layer DeepST models as the optimization structures to compare multi-step ahead traffic flow prediction.

Fig. 14 illustrates the results of traffic flow prediction for varying steps. As can be seen in Fig. 14, the RMSE as a whole indicated an upward trend with an increase in the number of steps. For the 30 min prediction, the value of the RMSE using the FDCN was 0.3336, which was 1.4% of the RMSE using the DeepST method. In addition, the prediction performance of the FDCN method was relatively stationary over a short period of time. This phenomenon implied that the proposed FDCN method could predict traffic flow more accurately than its competition methods and obtained improved prediction performance when compared with the ARIMA, DeepST, CNN and FCNN methods.

6. Conclusions

In the current study, we developed the FDCN model to predict traffic flow. The proposed method incorporated the fuzzy method and the deep residual convolution network to extract features for more accurate traffic flow prediction. We evaluated the FDCN model on the data set TaxiBJ, and verified performances of the FDCN. Experimental results indicated that the FDCN method was more powerful in its representation capability than existing methods. The FDCN has many open problems, including the optimized structure of the model and the influence of external factors on predictive performance. In the future, it will be interesting to consider reinforcement learning with a variety of external factors in traffic flow prediction.

Acknowledgments

This study was partially supported by the National Key Research and Development Plan of China under Grant No. 2016YFB0200405, the National Natural Science Foundation of China with Grant Nos. 61672217, 61370097, 61602164, and 61702172, and the Natural Science Foundation of Hunan Province, China under Grant 2018JJ2063 and 2018JJ3076.

References

[1] G.D.L. Torre, P. Rad, K.-K.R. Choo, Driverless vehicle security: challenges and future research opportunities, Future Gener. Comput. Syst. (2018). http://dx.doi.org/10.1016/j.future.2017.12.041. URL http://www.sciencedirect.com/science/article/pii/S0167739X17315066.

[2] J. Liu, X. Yu, Z. Xu, K.-K.R. Choo, L. Hong, X. Cui, A cloud-based taxi trace mining framework for smart city, Softw. Pract. Exp. 47 (8) (2016) 1081–1094.

[3] M.R. Jabbarpour, H. Zarrabi, R.H. Khokhar, S. Shamshirband, K.K.R. Choo, Applications of computational intelligence in vehicle traffic congestion problem: a survey, Soft Comput. (2017) 1–22.

[4] A. Koesdwiady, R. Soua, F. Karray, Improving traffic flow prediction with weather information in connected cars: a deep learning approach, IEEE Trans. Veh. Technol. 65 (12) (2016) 9508–9517.

[5] G. Xie, G. Zeng, Z. Li, R. Li, K. Li, Adaptive dynamic scheduling on multifunctional mixed-criticality automotive cyber-physical systems, IEEE Trans. Veh. Technol. 66 (8) (2017) 6676–6692.

[6] G. Xie, G. Zeng, J. Jiang, C. Fan, R. Li, K. Li, Energy management for multiple real-time workflows on cyber-physical cloud systems, Future Gener. Comput. Syst. (2017).

[7] W. Chen, J. An, R. Li, W. Li, Review on deep-learning-based cognitive computing, Acta Automat. Sinica 43 (11) (2017) 1886–1897.

[8] Y. Lv, Y. Duan, W. Kang, Z. Li, F.Y. Wang, Traffic flow prediction with big data: a deep learning approach, IEEE Trans. Intell. Transp. Syst. 16 (2) (2015) 865–873.

[9] J. An, G. Wen, W. Xu, Improved results on fuzzy h^{∞} filter design for t-s fuzzy systems, Discrete Dyn. Nat. Soc. 2010 (1) (2010) 9–11.

[10] X. Kong, Z. Xu, G. Shen, J. Wang, Q. Yang, B. Zhang, Urban traffic congestion estimation and prediction based on floating car trajectory data, Future Gener. Comput. Syst. 61 (2016) 97–107.

[11] S. Zhou, Q. Chen, X. Wang, Fuzzy deep belief networks for semi-supervised sentiment classification, Neurocomputing 131 (2014) 312–322.

[12] Y. Deng, Z. Ren, Y. Kong, F. Bao, Q. Dai, A hierarchical fused fuzzy deep neural network for data classification, IEEE Trans. Fuzzy Syst. (2016).

[13] J. An, L. Xinzhi, G. Wen, Stability analysis of delayed takagi-sugeno fuzzy systems: a new integral inequality approach, J. Nonlinear Sci. Appl. 10 (4) (2017) 1941–1959.

[14] C.P. Chen, C.-Y. Zhang, L. Chen, M. Gan, Fuzzy restricted boltzmann machine for the enhancement of deep learning, IEEE Trans. Fuzzy Syst. 23 (6) (2015) 2163–2173.

[15] J. Zhang, Y. Zheng, D. Qi, Deep spatio-temporal residual networks for citywide crowd flows prediction, CoRR abs/1610.00081, arXiv:1610.00081, 2016.

[16] J. Abdi, B. Moshiri, B. Abdulhai, A.K. Sedigh, Short-term traffic flow forecasting: parametric and nonparametric approaches via emotional temporal difference learning, Neural Comput. Appl. 23 (1) (2013) 141–159.

[17] S. Oh, Y.-J. Byon, K. Jang, H. Yeo, Short-term travel-time prediction on highway: a review of the data-driven approach, Transp. Rev. 35 (1) (2015) 4–32.

[18] S. Jin, D.-h. Wang, C. Xu, D.-f. Ma, Short-term traffic safety forecasting using gaussian mixture model and kalman filter, J. Zhejiang Univ. Sci. A 14 (4) (2013) 231–243.

[19] Q.-J. Kong, Q. Zhao, C. Wei, Y. Liu, Efficient traffic state estimation for large-scale urban road networks, IEEE Trans. Intell. Transp. Syst. 14 (1) (2013) 398–407.

[20] H. Chang, Y. Lee, B. Yoon, S. Baek, Dynamic near-term traffic flow prediction: system-oriented approach based on past experiences, IET Intel. Transport Syst. 6 (3) (2012) 292–305.

[21] S. Sun, C. Zhang, G. Yu, A bayesian network approach to traffic flow forecasting, IEEE Trans. Intell. Transp. Syst. 7 (1) (2006) 124–132.

[22] Y.S. Jeong, Y.J. Byon, M.M. Castro-Neto, S.M. Easa, Supervised weighting-online learning algorithm for short-term traffic flow prediction, IEEE Trans. Intell. Transp. Syst. 14 (4) (2013) 1700–1707.

[23] K. Kumar, M. Parida, V.K. Katiyar, Short term traffic flow prediction for a non urban highway using artificial neural network, Procedia - Soc. Behav. Sci. 104 (2013) 755–764.

[24] H. Su, S. Yu, Hybrid ga based online support vector machine model for short-term traffic flow forecasting, Adv. Parallel Process. Techn. (2007) 743–752.

[25] W. Hong, Y. Dong, F. Zheng, S. Wei, Hybrid evolutionary algorithms in a svr traffic flow forecasting model, Appl. Math. Comput. 217 (15) (2011) 6733–6747.

[26] S. Dunne, B. Ghosh, Regime-based short-term multivariate traffic condition forecasting algorithm, J. Transp. Eng. 138 (4) (2011) 455–466.

[27] W. Min, L. Wynter, Real-time road traffic prediction with spatio-temporal correlations, Transp. Res. C 19 (4) (2011) 606–616.

[28] W. Huang, G. Song, H. Hong, K. Xie, Deep architecture for traffic flow prediction: deep belief networks with multitask learning, IEEE Trans. Intell. Transp. Syst. 15 (5) (2014) 2191–2201.

[29] P. Pongpaibool, P. Tangamchit, K. Noodwong, Evaluation of road traffic congestion using fuzzy techniques, in: TENCON 2007-2007 IEEE Region 10 Conference, IEEE, 2007, pp. 1–4.

[30] X. Zhang, E. Onieva, A. Perallos, E. Osaba, V.C. Lee, Hierarchical fuzzy rule-based system optimized with genetic algorithms for short term traffic congestion prediction, Transp. Res. C 43 (2014) 127–142.

[31] H. Shankar, P. Raju, K.R.M. Rao, Multi model criteria for the estimation of road traffic congestion from traffic flow information based on fuzzy logic, J. Transp. Techn. 2 (1) (2012) 50–62.

[32] P. Lopez-Garcia, E. Onieva, E. Osaba, A.D. Masegosa, A. Perallos, A hybrid method for short-term traffic congestion forecasting using genetic algorithms and cross entropy, IEEE Trans. Intell. Transp. Syst. 17 (2) (2016) 557–569.
[33] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, 2015, 770–778.
[34] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, A.Y. Ng, Multimodal deep learning, in: Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011, pp. 689–696.
[35] N.K. Kasabov, Q. Song, Denfis: dynamic evolving neural-fuzzy inference system and its application for time-series prediction, IEEE Trans. Fuzzy Syst. 10 (2) (2002) 144–154.
[36] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, Comput. Sci. (2014).
[37] J. Zhang, Y. Zheng, D. Qi, R. Li, X. 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.

Weihong Chen is a Ph.D. candidate at the College of Computer Science and Electronic Engineering, Hunan University, and a professor at Hunan City University. She received her Master's degree from Hunan University in 2006. Her research interest covers cyber–physical systems, distributed computing, and machine learning.

Jiyao An received the M.Sc. degree in Mathematics from Xiangtan University, China, the Ph.D. degree in Mechanical Engineering from Hunan University, China, in 1998, and 2012, respectively. He was a Visiting Scholar with the Department of Applied Mathematics, University of Waterloo, Ontario, Canada, from 2013 to 2014. Since 2000, he joined the College of Computer Science and Electronic Engineering in Hunan University, Changsha, China, where he is currently a full Professor. His research interests include Automotive cyber–physical Systems (ACPS), Takagi–Sugeno fuzzy systems, Parallel and Distributed Computing, and Computational Intelligence for Big Data. He has publish more than 60 papers in international and domestic journals and refereed conference papers. He is a member of the IEEE and ACM, and a senior member of CCF. He is an active reviewer of international journals.

Renfa Li is a Professor of computer science and electronic engineering, and the Dean of College of Computer Science and Electronic Engineering, Hunan University, China. He is the Director of the Key Laboratory for Embedded and Network Computing of Hunan Province, China. His major interests include computer architectures, embedded computing systems, cyber–physical systems, and Internet of things. He is a member of the council of CCF, a senior member of IEEE, and a senior member of ACM.

Guoqi Xie received his Ph.D. degree in computer science and engineering from Hunan University, China, in 2014. He was a Postdoctoral Researcher at Nagoya University, Japan, from 2014 to 2015. Since 2015 He is working as a Postdoctoral Researcher at Hunan University, China. He has received the best paper award from ISPA 2016. His major interests include embedded and real-time systems, parallel and distributed systems, software engineering and methodology. He is a member of IEEE, ACM, and CCF.

Md. Zakirul Alam Bhuiyan received the B.Sc. degree in computer science and technology from the International Islamic University Chittagong, Kumira, Bangladesh, in 2005, and the M.Eng. and Ph.D. degrees in computer science and technology from Central South University, Changsha, China, in 2009 and 2013, respectively.

He is currently an Assistant Professor with the Department of Computer and Information Sciences, Fordham University, Bronx, NY, USA. His research focuses on dependable cyber–physical systems, emerging wireless network applications, big data, cloudcomputing, and cyber security. He has over 100 publications that have appeared in the prestigious journals/conferences in the domain, including the IEEE TII, IEEE TC, IEEE TPDS, and so on. He is a member of the ACM. He has received the IEEE TCSC Award for Excellence in Scalable Computing for Early Career Researchers (2016–2017), the IEEE Outstanding Leadership Award (2016) and Service Award (2017), the Young Scientist Funding Award, China, the Provincial Best Ph.D. Thesis Award, and the Best Paper Awards (IEEE MASS 2016 and IEEE ISPA 2013). He has served as the general chair, program chair, workshop chair, publicity chair, TPC member, and a reviewer of various international journals/conferences. He is currently the General Chair for IEEE DASC 2018, Greece, and DependSys 2018, China, and the Program Chair for IEEE I-SPAN 2018, China, and IEEE SmartWorld 2018, China, and a TPC Member of IEEE INFOCOM 2018, USA. He has also served as an Associate/Lead Guest Editor for key journals, including the IEEE TRANSACTIONS ON BIG DATA, the ACM Transaction on cyber–physical Systems, Information Sciences, the IEEE INTERNET OF THINGS JOURNAL, and FGCS.

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 cyber–physical systems. He has published over 550 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.