# A Data Grouping CNN Algorithm for Short-term Traffic Flow Forecasting

Donghai Yu[1], Yang Liu[1, ⋆], and Xiaohui Yu[1,2]

[1] School of Computer Science and Technology, Shandong University, Jinan, China, 250101
[2] School of Information Technology, York University, Toronto, ON, Canada, M3J 1P3
yliu@sdu.edu.cn, xyu@sdu.edu.cn

**Abstract.** In this paper, a data grouping approach based on convolutional neural network(DGCNN) is proposed for forecasting urban short-term traffic flow. This approach includes the consideration of spatial relations between traffic locations, and utilizes such information to train a convolutional neural network for forecasting. There are three advantages of our approach: (1)the spatial relations of traffic flow are adopted; (2)high-quality features are extracted by CNN; and (3)the accuracy of forecasting short-term traffic flow is improved. To verify our model, extensive experiments are performed on a real data set, and the result shows that the model is more effective than other existing methods.

**Keywords:** Convolution Neural Network, Traffic Flow Forecasting, CBOW, Deep Learning

## 1 Introduction

Traffic problems are crucial issues in the rapidly developing society. Traffic flow forecasting can be an important problem in Intelligent Transportation System(ITS) for urban construction. To build the Wisdom City, forecasting traffic flow in real time is a realistic demand. In a modern city, the change of traffic flow has a profound impact on people's daily life, such as the route selection for drivers. Long-term traffic flow forecasting usually refers to predicting the traffic by month or year in advance. Due to unpredictable disruptions, long-term forecasting may not be accurate enough for practical use [1]. Therefore, it might be more helpful to make a shot-term prediction, e.g., we may just want to know the traffic flow volume within 10 minutes to decide our route while driving.

A variety of techniques have been applied in the context of short-term traffic flow forecasting, including moving average methods [19], k-nearest-neighbor methods [6], auto regressive MA (ARIMA) or seasonal ARIMA (SARIMA) model [20, 21], and neural networks (NNs) [7, 15]. Forecasting traffic flow heavily depends on historical and real-time traffic data collected from various sensor sources, such as inductive loops, radars, cameras, mobile Global Positioning System, social media, etc.

⋆ Corresponding author.

Despite the popularity of this research, existing work tends to suffer from the following problems. Firstly, in the downtown area of a city where traffic jam occurs, the traffic flows at different locations will influence each other. In a complicated traffic network, the influences between different locations are decided by traffic network structure, signal light, regionalism, commercial layout, etc. Previous methods just built the model based on time-series data regardless of spatial influences. Secondly, most existing studies only adopt few useful features, which can not provide sufficient information for forecasting.

In recent years, deep learning has drawn a lot of academic and industrial attention [2], and it has been applied with success in the tasks of classification, natural language processing, dimensionality reduction, object detection, motion modeling, etc. There are also early attempts to apply deep learning to traffic flow forecasting. For example, using SAE [15] and using deep belief networks [10]. Both methods first use the deep network to rebuild input features, and then train regression network for forecasting the traffic flow. However, neither has considered the traffic impact among different locations.

In this paper, We propose a Data Grouping Convolutional Neural Network (DGCNN) approach. In this method, to solve above problems, we apply Convolutional Neural Network(CNN) to forecast traffic flow and use CBOW model to find the spatial influences that exist in locations. In DGCNN, the inputs are the historical traffic flow data from the target location and historical data of other locations. We group these data to build feature matrix for forecasting. To discover the traffic flow relations at different locations, we calculate the location vectors by CBOW model [16], and we expect that the distances between location vectors can reflect each relation.

The contributions of this paper can be summarized as follows.

1)We propose a Data Grouping Convolutional Neural Networks to forecast short-term traffic flow. To the best of our knowledge, it is the first time that the CNN has been applied to traffic flow forecasting.

2)We identify a spatial relationship between different locations in the city by calculating the distance between location vectors, and utilize the relationship to increase the accuracy of traffic flow forecasting.

3)We perform extensive experiments using a real data set and the results demonstrate the effectiveness of DGCNN.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 provides a brief description of preliminary preparations. Section 4 designs and describes the DGCNN Modeling for the short-term traffic flow forecasting. In Section 5, the experiment details are discussed. Finally, the conclusions and future directions, regarding short-term traffic flow predictor design using the DGCNN approach, are presented in Section 6.

## 2 Related work

Researchers have been trying to make traffic flow forecasting more accurate since 1970s [15]. Based on the length of time interval, there are long-term forecasting and short-term forecasting. Based on the parameter mode, there are parametric methods, non-parametric models, and simulation methods. In the following, we will introduce some literatures which are mostly related to our work.

Long-term forecasting indicates making a prediction by month or by year. The volume of traffic flow is large and relatively stable, and it is slightly affected by daily accident. For example, Papagiannaki et al. [12] proposed a method based on ARIMA to forecast network traffic flow in a month after they revealed that their time series data had the long term trend and the fluctuations at the 12 hour time scale.

For short-term forecasting, auto-regressive integrated moving average(ARIMA) [5] models and artificial neural network(ANN) [13] models are are widely exploited. Lv et al. [14] proposed a plan moving average algorithm for utilizing previous days' historical data. In addition, there are some integration models for this task. For example, Chang et al. [4] proposed a 3-stage model which integrates ARIMA and ANN, witch uses the ARIMA forecasting data as a part of input of ANN. Moretti et al. [17] studied an ensemble model which consists of a statistical method and a neural network bagging model. Besides, researchers also consider the use of integrated data. As another example, Oh et al. [18] proposed a Gaussian mixture model clustering(GMM) method to partition the data set for training ANN. Deep learning methods have also gained a lot of attention recently. Lv et al. [15] used a stacked autoencoder(SAE) to learn generic traffic flow features. Huang, Song and Huang et al. [10] applied a deep belief networks model in traffic flow prediction, which adopts multitask learning to reduce the error.

There are also some simulation methods in the same vein. For example, Fusco et al. [9] proposed a quasi-dynamic traffic assignment model to simulate the traffic flow. Fusco et al. [8] applied network-based machine learning models and dynamic traffic assignment models on a large urban traffic network. Abadi [1] used a simulation system to solve the problem of data sparseness where the simulation system produces simulative traffic data to supplement the traffic flow data set.

## 3 Preliminaries

In this section, we will define a few terms that are required for the subsequent discussion, and describe our modeling strategy for the vehicle traffic flow forecasting.

**Definition 1 (Traffic Flow).** *For each location, we record the total number of vehicles that have past during a certain period of time. Traffic flow is defined as* $f_{l_i}^{d}(n)$ *where* $l_i$ *is the location ID,* $d$ *is the date and* $n$ *stands for n-th period(e.g., when the period is set to 10 minutes,* $f_0^{0201}(0)$ *is the traffic flow at location whose id is* 0 *in February 1st from 00:00 to 00:10 ).*

**Definition 2 (Trajectory).** *For each vehicle, trajectories are collected by recording the locations the vehicle has passed sequentially. A trajectory* $T_p$ *is defined* *as a series of location IDs as follows.*

$$T_p : \{l_0, l_1, l_2, ..., l_{s-1}, l_s\}$$

*Here, p is the ID of a vehicle,* $l_0, l_1, ...., l_s$ *are the IDs of the locations.* $\{l_0, l_1, ...., l_s\}$ *is ordered by passing time, and s is the length of the trajectory(e.g., if a vehicle of which the ID is* 0001 *has orderly passed location* $l_a$*,* $l_b$*,* $l_c$ *and* $l_d$*, here is a Trajectory* $T_{0001} : \{l_a, l_b, l_c, l_d\}$ *).*

## 4 DGCNN Modeling

In this paper, we propose a novel DGCNN model to forecast the short-term traffic flow. In order to develop a robust model for the traffic flow prediction, we first need to accumulate a sufficient amount of training data that would well explain the traffic patterns. However, it is clear that for each single spot, such traffic flow data fade out rapidly, leaving low impact on its future traffic flow prediction. Meanwhile, we notice that in a dense traffic network, the traffic flows at different locations may have strong relationship between each other. For example, if there are no exits on the road between location $l_a$ and $l_b$, the traffic flows of $l_a$ and $l_b$ would be similar. So, the relationships in the traffic network are also considered in our model.

Multi-layers Convolutional Neural Network(CNN) contains convolution layer, pooling layer, loss-function layer, etc. Based on different types of loss-function layers, we can perform different tasks such as classification and regression. With the assist of convolution layer and pooling layer, CNN can help understanding the relations between the features. Properly handled, CNN can be a powerful tool for modeling the future traffic flow.

CBOW model [16] has been applied in natural language processing, with its capacity to represent the word in vector with more comprehensive information. In the CBOW model, the more co-occurences two words have, the more similar patterns they share, and the larger similarity values the two vectors have. Motivated by this idea, we aim to find the spatial relations of traffic flow from trajectory data. When the traffic flow at different locations has strong relation among each other, therefore can be grouped together. Further, to address the problem that arises from data sparsity, we can borrow features of similar locations to train a more robust CNN for flow prediction.

There are two stages in training DGCNN: grouping and training. In the first part, we use the vectors to represent the locations so that we can find every location's k-nearest locations by evaluating their similarities, e.g., using a standard Euclidean Distance. The output of this first step is the grouping result of these locations. In the second part, we will train a deep convolutional neural networks to forecast traffic flow. The input features for the network comes from the locations which are in the same group.

### 4.1 Grouping

In order to discover locations that are similar to the target, we first need to gain a comprehensive understanding the location, and represent it in a proper way. Word2vec can be used for learning high-quality word vectors from huge data sets with billions of words when Word2vec has implemented the CBOW model based on Negative Sampling [16]. In the task of prediction, we regard the locations in trajectories as words while trajectories represent documents.

CBOW is a 3-layer neural network which includes an input layer, a projection layer, and an output layer.

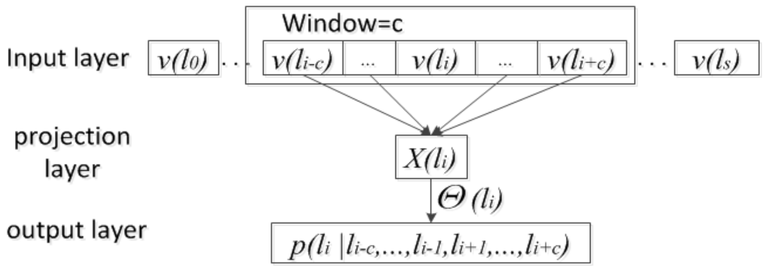

**Fig. 1.** CBOW model

Fig.1. shows the structure of the CBOW model, where $v(l_i)$ is the location vector for $l_i$, $\theta(l_i)$ is the parameter vector, and Equation (1) and (2) indicate the outputs of projection layer and output layer respectively.

$$X(l_i) = \sum_{x=i-c}^{i-1} v(l_x) + \sum_{x=i+1}^{i+c} v(l_x) \tag{1}$$

$$p(li|l_{i-c}, ..., l_{i-1}, l_{i+1}, ..., l_{i+c}) = \frac{1}{1 + e^{-X^{\mathrm{T}}(l_i)\theta(l_i)}} \tag{2}$$

In order to reduce the cost of training, the strategy of negative sampling is adopted. When we train the vector of $l_i$, we can get a corresponding negative sample set $NEG(l_i)$. If $l_x \in NEG(l_i)$, then $l_x \neq l_i$. Mikolov et al. [16] introduces the detailed method to find negative samples. The detailed process of updating system parameters are shown in Equation (3) and (4). Here, $\eta$ is the learning rate.

$$L^{l_i}(l_x) = \begin{cases} 0, l_x = l_i; \\ 1, l_x \neq l_i; \end{cases}$$

$$\theta(l_x) := \theta(l_x) + \eta[L^{l_i}(l_x) - \frac{1}{1 + e^{-X^{\mathrm{T}}(l_i)\theta(l_x)}}]X(l_i), l_x \in NEG(l_i) \tag{3}$$

$$v(l_x) := v(l_x) + \eta \sum_{l_x \in \{l_i\} \bigcup NEG(l_i)} [L^{l_i}(l_x) - \frac{1}{1+e^{-X^{\mathrm{T}}(l_i)\theta(l_x)}}]\theta(l_x), \quad l_x \in \{l_{i-c}, ..., l_{i-1}, l_{i+1}, ..., l_{i+c}\} \tag{4}$$

The input for word2vec are all the trajectories $\{T_p,\ p \in V\}$ where $V$ is the set of vehicle IDs. The output are the key-value tuples. For each tuple, the key is the location ID and the value is a $w$ dimension vector. The structure of a tuple can be represented as follows:

$$< l_i \ : \ v(l_i) > \ \ (v(l_i) \in \mathbb{R}^w)$$

In this way, each location can be represented as a vector. We use the Euclidean Distance between the vectors to represent closeness between locations.

$$distance(l_i, l_j) = ||v(l_i) - v(l_j)|| \tag{5}$$

For each location $l_a$, we calculate the distances to other locations and sort out the result. Further, we select the top-k locations $\{l_b, l_c, l_d, \cdots\}$, where k is usually less than 5. Then, location $l_a$ and the top-k locations can be grouped together in group $G_a : \{l_a, l_b, l_c, l_d, \cdots\}$. Finally, we obtain $k+1$ locations in group $G_a$, and the locations after $l_a$ are ordered by their distances to $l_a$.

## 4.2 Feature Matrix

To integrate the temporal-spatial traffic flow information into the input, we build a feature matrix. Note that to provide a more comprehensive understanding of traffic flow, not only the traffic data, but also its mean value and the mode are considered as the elements of the input matrix.

**Average traffic flow** For every location, as the traffic flow is relatively stable at the same time of a day, we calculate the mean value for historical traffic flow data. For a historical volume $f^d_{l_i}(n)$, the corresponding average traffic flow $m^d_{l_i}(n)$ is calculated as follows:

$$m^d_{l_i}(n) = \frac{1}{7} \sum_{a=d-1}^{d-7} f^a_{l_i}(n) \tag{6}$$

Here, we choose a previous week's mean value to represent the average traffic flow.

**Mode** During the process of prediction, outliers can make the average traffic flow volume abnormal. To eliminate such effect, we include the mode $\mathrm{r}^d_{l_i}(\mathrm{n})$ for $f^d_{l_i}(n)$. However, as the traffic flow volumes are continuous, they need to be discretized before use as input. Accordingly, it can be evaluated as follows:

$$r^d_{l_i}(n) = mode(\lfloor \frac{f^{d-1}_{l_i}(n)}{\beta} \rfloor \times \beta, \lfloor \frac{f^{d-2}_{l_i}(n)}{\beta} \rfloor \times \beta, ..., \lfloor \frac{f^{d-7}_{l_i}(n)}{\beta} \rfloor \times \beta) \tag{7}$$

Here, $\beta$ is an integer which is smaller than the maximum value of traffic flow, and the mode is calculated according to the result of the previous week. In practice, the best $\beta$ can be data dependent. We vary the value of $\beta$ from 1 to 10, and notice that the best result can be obtained at 4 when the average traffic flow is smaller than 100, otherwise $\beta$ is 10. We apply such setting in the rest of the paper.

We build a traffic flow feature matrix using all the three types of traffic flow data: the real data, the average data, and the mode data. besides the three types data, we combine the traffic flow features that comes from the locations which in a same group. For target $\mathrm{f}_{l_0}^d(\mathrm{x})$, the following matrix shows the configuration of the feature matrix.

$$\begin{bmatrix} f_{l_0}^d(x-p) & ... & f_{l_0}^d(x-2) & f_{l_0}^d(x-1) \\ m_{l_0}^d(x-p) & ... & m_{l_0}^d(x-2) & m_{l_0}^d(x-1) \\ r_{l_0}^d(x-p) & ... & r_{l_0}^d(x-2) & r_{l_0}^d(x-1) \\ & & & \\ f_{l_1}^d(x-p) & ... & f_{l_1}^d(x-2) & f_{l_1}^d(x-1) \\ m_{l_1}^d(x-p) & ... & m_{l_1}^d(x-2) & m_{l_1}^d(x-1) \\ r_{l_1}^d(x-p) & ... & r_{l_1}^d(x-2) & r_{l_1}^d(x-1) \\ . & ... & . & . \\ . & ... & . & . \\ f_{l_k}^d(x-p) & ... & f_{l_k}^d(x-2) & f_{l_k}^d(x-1) \\ m_{l_k}^d(x-p) & ... & m_{l_k}^d(x-2) & m_{l_k}^d(x-1) \\ r_{l_k}^d(x-p) & ... & r_{l_k}^d(x-2) & r_{l_k}^d(x-1) \end{bmatrix}$$

Here, $p$ is number of previous period, and it should be smaller than x, and $l_0, l_1, , , , l_k$ are in the same group of $G_0$. The size of the matrix is decided by $p$ and the group size $k + 1$. In horizontal and vertical directions, the elements are sorted by time and location relation. Therefore, int the matrix, the spatial impacts can be reflected in the vertical direction and the temporal influences are extracted in the horizontal direction.

## 4.3 Training

In the second part of DGCNN, to use the 2-dimensional temporal-spatial traffic flow features for forecasting, we adopt the Convolutional Neural Network(CNN) with its capacity to process the temporal-spatial features at the same time.

CNN is built by convolution layers, sub-sampling layers, and full connected layers, etc. It stands with an array of alternating convolution and sub-sampling operations, and then continues with generic multi-layer network. The last few layers (closest to the outputs) are fully connected with 1-dimensional layers [3] for regression analysis. Fig.2. shows the structures of the convolution layer, and Fig.3 is the sub-sampling layer.

The input and output of convolution layers are both feature maps. The active function that used in the convolution layer is shown in Equation (8). $M_j$ indicates the kernel $j$'s projection positions on the feature map, $k_{ij}^l$ is the kernel weights,

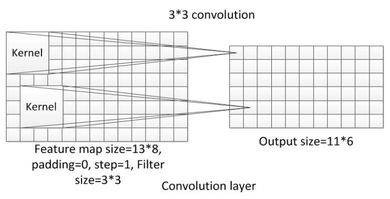

**Fig. 2.** The structures of convolution layer

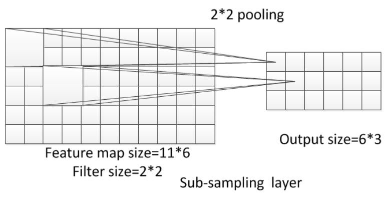

**Fig. 3.** The structures of sub-sampling layer

$b_j$ is a bias for kernel $j$, $x_j^l$ and $x_i^{l-1}$ are the volumes of output and input feature maps' elements, and $f(\sim)$ is a *sigmoid* function.

$$x_j^l = f(\sum_{i \in M_j} x_i^{l-1} \times k_{ij}^l + b_j^l) \tag{8}$$

Equation (9) shows the sub-sampling layer function. Different from convolution layers, the $pool(\sim)$ is a sampling function, usually using the maximum or average value as output. Note the matrices on the feature map are mutually disjoint, and Fig. 3 shows this character.

$$x_j^l = f(pool(x_j^{l-1}) + b_j^l) \tag{9}$$

Apparently, we can see that the sub-sampling layer would reduce the feature dimensions, and the convolution layer can extend the feature dimension by increasing the number of kernels. In this study, we choose 2 convolution layers with 15 kernels and 7 kernels, and 1 sub-sampling layer. Table. 1 shows an example whose input matrix size is set respectively to be $k = 1, p = 20$. Note that in the last layer, there is only one node, and its task is to apply regression for prediction. Hence, the output of CNN is the forecasting value.

**Table 1.** CNN layers' size

| No. | type | input size | output size |
|---|---|---|---|
| 1 | data | 1 * 6 * 20(120) | 1 * 6 * 20(120) |
| 2 | convolution | 1 * 6 * 20(120) | 15 * 4 * 8(1080) |
| 3 | pool(sub-sampling) | 15 * 4 * 8(1080) | 15 * 2 * 9(270) |
| 4 | convolution | 15 * 2 * 9(270) | 7 * 1 * 8(56) |
| 5 | full-connection | 7 * 1 * 8(56) | 300 |
| 6 | full-connection | 300 | 1 |
| 7 | EuclideanLoss | 1 | 1 |

## 5 Experiment

### 5.1 Data

The dataset used in the experiments consist of real vehicle passage records from December 1, 2015 to February 29, 2016 that were collected from the traffic surveillance system of a major metropolitan area. The dataset contains 479,451,848 passage records, and involves a total of 1323 detecting locations on the main roads. The time interval for counting traffic flow volume is 12 minutes. For every location, we divide the traffic flow data set into two parts, a training set (80%) and a testing set (20%).

### 5.2 Evaluation Matrics

We use two widely employed evaluation measures to assess the forecast performance:

1) The Root Mean Squared Error (RMSE) is a way to measure the average error of the forecasting results and is calculated by

$$RMSE = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(y_n - \hat{y}_n)^2}$$

Here, y$_n$ and $\hat{y}_n$ are the observed and the forecast values of observation n. N is the total number of locations.

2) The Mean Relative Error (MRE) is a way to measure the proportional error of the forecasting results and is calculated by

$$MRE = \frac{1}{N}\sum_{n=1}^{N}\frac{|y_n - \hat{y}_n|}{y_n} \times 100\%$$

### 5.3 Evaluation of DGCNN

There are two steps in the experiment. Firstly, trajectory data are used to train the word2vec, and we find every location's $k$ nearest other locations. Secondly, the feature matrix is fed into CNN for prediction.

In the first step, we get 1,592,562 trajectories for training the word2vec. To generate a robust model, we ignore locations which appear less than 5 times. For one location in a trajectory, we consider that its previous 3 and next 3 locations as the impact factors($window$ = 3). One iteration is not enough for training high-quality location vectors, so $iteration$ can be more than one, Here we set it as 5. To make the vectors more effective, we set the vector dimension to be 350.

In the second step, We employ Caffe [11] to build the CNN. Different values of $p$ (the height of feature matrix) and $k$(the width of feature matrix) are tested in the experiment. We want to minimize the MRE while training the CNN. Fig.4. shows the MREs of CNN with different values of group size $k+1$ and Fig.5 shows MREs with different previous interval number $p$. Two group of $G_1$ and $G_2$ have been tested.

From Fig.4 and Fig.5, we find that $k = 1, p = 20$ are the best values for DGCNN.

## 5.4 Comparison of Results

We compare SingleCNN($k = 0, p = 20$) with DGCNN($k = 1, p = 20$), Fig.6. shows the RMSEs with different iterations and Fig.7 shows the MREs. The result demonstrates that the grouped data can effectively decrease the MRE and RMSE and improve the performance.

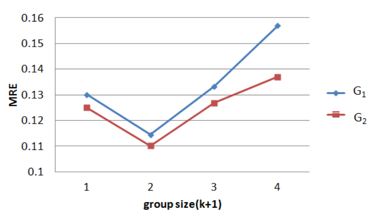

**Fig. 4.** MREs with different values of $k + 1$

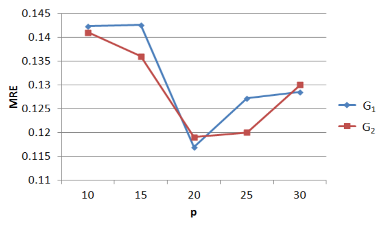

**Fig. 5.** MREs with different values of $p$

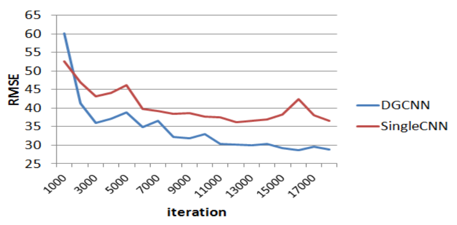

**Fig. 6.** RMSEs for SingleCNN and DGCNN

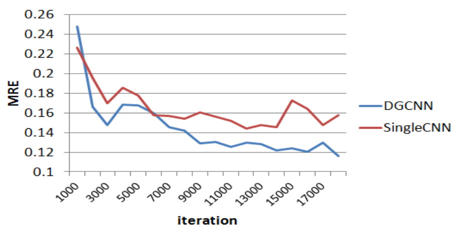

**Fig. 7.** MREs for SingleCNN and DGCNN

We also compare other models' forecasting results with DGCNN. These models include MA, ARIMA and SAE.

**1) MA Model:** An MA of order is computed by

$$\hat{y}_{t+1} = \frac{y_t + y_{t-1} + y_{t-2} + ... + +y_{t-k+1}}{k},$$

where $k$ is the number of terms in the MA. The MA technique deals only with the latest $k$ periods of known data, and the number of data points in each average dose not change as time continues.

**2) ARIMA Model:** We adopt a general ARIMA model of order $(r, d, s)$, where $d$ is the order of differencing, and orders $r$ and $s$ are the AR and MA operators.

**3) SAE Model:** SAE model uses stacked autoencoder to extract features, and trains a regression network for forecasting.

Fig.8. shows their average MREs, and Fig.9 reports their average RMSEs. All these results demonstrate that DGCNN model can make a more accurate short-term traffic flow forecasting.

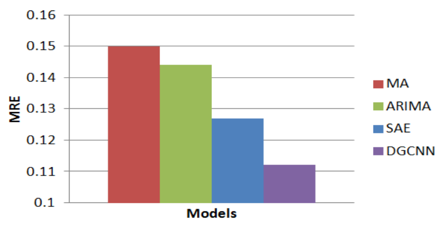

**Fig. 8.** MRE of different models in comparison

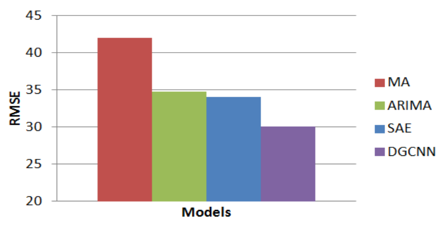

**Fig. 9.** RMSE of different models in comparison

## 6 Conclusion

In this paper, we propose a DGCNN model for forecasting traffic flow. Different to previous models, which have paid most of the attention on time-series data, this model uses a spatio-temporal dataset to build 2 networks. A CBOW model is used to find the spatio-relations, and the CNN is used to forecast traffic flow when the input is a spatio-temporal data matrix. We have compared DGCNN model with some state-of-the-art methods including MA, ARIMA and SAE. The result shows that the DGCNN can make more accurate traffic flow prediction than other models.

For further work, it is extraordinarily meaningful to combine the two stages of DGCNN for the forecasting task. We will extend our work on other types of input features, and we want to optimize the network structure for a better result.

## 7 Acknowledgment

This work was supported in part by the National Basic Research 973 Program of China under Grant No. 2015CB352502, the National Natural Science Foundation of China under Grant Nos. 61272092 and 61572289, the Natural Science Foundation of Shandong Province of China under Grant Nos. ZR2012FZ004 and ZR2015FM002, the Science and Technology Development Program of Shandong Province of China under Grant No. 2014GGE27178, and the NSERC Discovery Grants.

## References

1. Afshin Abadi, Tooraj Rajabioun, and Petros /A/. Ioannou. Traffic flow prediction for road transportation networks with limited traffic data. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):653–662, 2015.
2. Yoshua Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.
3. Jake Bouvrie. Notes on convolutional neural networks. *Neural Nets*, 2006.
4. S. C. Chang, R. S. Kim, S. J. Kim, and M. H. Ahn. Traffic-flow forecasting using a 3-stage model. In *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*, pages 451–456, 2000.

5. Chenyi Chen, Jianming Hu, Qiang Meng, and Yi Zhang. Short-time traffic flow prediction with arima-garch model. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 607–612, 2011.
6. Gary A Davis and Nancy L Nihan. Nonparametric regression and short-term freeway traffic forecasting. *Journal of Transportation Engineering*, 117(2):178–188, 1991.
7. Hussein Dia. An object-oriented neural network approach to short-term traffic forecasting. *European Journal of Operational Research*, 131(2):253–261, 2001.
8. G. Fusco, C. Colombaroni, L. Comelli, and N. Isaenko. Short-term traffic predictions on large urban traffic networks: Applications of network-based machine learning models and dynamic traffic assignment models. In *International Conference on MODELS and Technologies for Intelligent Transportation Systems*, 2015.
9. Gaetano Fusco, Chiara Colombaroni, Andrea Gemma, and Stefano Lo Sardo. A quasi-dynamic traffic assignment model for large congested urban road networks. *International Journal of Mathematical Models and Methods in Applied Sciences*, 7(1):63–74, 2013.
10. Wenhao Huang, Guojie Song, Haikun Hong, and Kunqing Xie. Deep architecture for traffic flow prediction: deep belief networks with multitask learning. *Intelligent Transportation Systems, IEEE Transactions on*, 15(5):2191–2201, 2014.
11. Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *Eprint Arxiv*, pages 675–678, 2014.
12. Papagiannaki Konstantina, Taft Nina, Zhang Zhi-Li, and Diot Christophe. Long-term forecasting of internet backbone traffic. *IEEE Transactions on Neural Networks*, 16(5):1110–1124, 2005.
13. Corinne Ledoux. An urban traffic flow model integrating neural networks. *Transportation Research Part C Emerging Technologies*, 5(5):287–300, 1997.
14. Lei Lv, Meng Chen, Yang Liu, and Xiaohui Yu. *A Plane Moving Average Algorithm for Short-Term Traffic Flow Prediction.* 2015.
15. Yisheng Lv, Yanjie Duan, Wenwen Kang, Zhengxi Li, and Fei-Yue Wang. Traffic flow prediction with big data: A deep learning approach. *Intelligent Transportation Systems, IEEE Transactions on*, 16(2):865–873, 2015.
16. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
17. Fabio Moretti, Stefano Pizzuti, Stefano Panzieri, and Mauro Annunziato. Urban traffic flow forecasting through statistical and neural network bagging ensemble hybrid modeling. *Neurocomputing*, 167:3–7, 2015.
18. Se Do Oh, Young Jin Kim, and Ji Sun Hong. Urban traffic flow prediction system using a multifactor pattern recognition model. *IEEE Transactions on Intelligent Transportation Systems*, 16(5):1–12, 2015.
19. Brian L Smith, Billy M Williams, and R Keith Oswald. Comparison of parametric and nonparametric models for traffic flow forecasting. *Transportation Research Part C: Emerging Technologies*, 10(4):303–321, 2002.
20. Billy Williams. Multivariate vehicular traffic flow prediction: Evaluation of arimax modeling. *Transportation Research Record Journal of the Transportation Research Board*, 1776(1):194–200, 2001.
21. Billy M Williams and Lester A Hoel. Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results. *Journal of transportation engineering*, 129(6):664–672, 2003.