Performance analysis of machine learning methods for short-term traffic prediction

Zhiheng Jiang[1, 4, †], **Jingyao Li**[2, †] **and Shixuan Liu**[3, †]

[1] School of Civil Aviation University, Tianjin, 618307, China
[2] School of Tongji University, Shanghai, 200092, China
[3] School of Xidian University, Xi'an, Shaanxi, 710126, China

[4]AlfredaLingLing@mail.dlut.edu.cn
[†] These authors contributed equally

Abstract. With the rapid development of urbanization and the rapid increase of the number of motor vehicles, the problem of urban traffic congestion has become increasingly prominent. The accurate prediction of short-term traffic flow is considered as a promising solution, which can provide a key decision-making basis for route planning and traffic flow scheduling, so that can greatly alleviate or even prevent congestion. Researchers have used many machine learning methods to predict traffic flow, but few people pay attention to the boundaries of different machine algorithms. In this paper, we use AdaBoost, Random Forest, SVM and BP neural network to predict short-term traffic flow in California, which aims to compare the differences in prediction performance of different algorithms and analyze their potential reasons. The results show that, the integration methods such as AdaBoost and Random Forest are quite appropriate to solve the short-term traffic flow, which can obtain an accuracy more than 95%, while prediction made by SVM is less precise than the two aforementioned methods with a 79% accuracy. And BP neural network may be inappropriate if the parameters remain default. The different results are due to the periodicity of the database. Integration methods can recognize the periodicity while the SVM and BP neural network fail to do it. When employing the SVM and BP neural network, the datasets need to be divided within a period to avoid being disturbed by cyclically. Besides, the precise of BP neural network can be improved when adjusting the parameters to the optimal.

Keywords: short-term traffic flow prediction, AdaBoost, Random Forest, SVM, BP neural network.

1. Introduction

With the increase of the number of private cars, traffic congestion has become increasingly prominent, which has become an inevitable research hotspot and difficulty in realizing intelligent transportation [1]. Congestion is a waste of money and experience. In the United States, the annual time loss caused by traffic congestion is about 5.5 billion hours, the annual fuel waste is more than 2.9 billion gallons, and the capital waste is more than 100 billion dollars [2]. The accurate traffic forecast shows a promise future in alleviating the traffic congestion, which has attracted more and more research attention.

Traffic flow prediction is the main component of intelligent transportation system (ITS). Intelligent transportation system aims to improve the efficiency of transportation infrastructure, which needs driver early warning system and future traffic information required for various control decisions. This relies on advanced models that can accurately predict traffic parameters [3]. When it comes to traffic flow prediction, it aims to predict the number of vehicles passing through the designated road section within the agreed time, which is also known as traffic flow prediction [4]. At present, there are various models to predict short-term traffic flow. The most representative method is based on statistical analysis. For example, the Autoregressive Integrated Moving Average model (ARIMA) is widely used. By analysing statistical time series, Ye et al realized passenger flow prediction in bus transportation system [4]. In addition, Kalman filtering has been applied successfully for demand prediction with great accuracy [5]. The Nonlinear theoretical models are more complex and more accurate. Other work applied the chaotic time series analysis to urban short-term traffic flow data, which obtained better accuracy [7]. Considering the existence of many mutation factors, Forbes introduced catastrophe theory [7].

With the development of AI, researchers used neural network to improve algorithms. A forecast method based on modified genetic algorithm (GA) optimized BP neural network was proposed [9], which can consider more influent factors but not suit real-time prediction. Some works used k-nearest nonparametric regression to forecast and some introduced GPR model which reduce interference of natural and human factors [10]. In addition, deep learning is an innovative method. Facing a large-scale, multi-dimension, nonlinear and non-normal distribution time series data, Support Vector Machine (SVM) regression algorithm [11], random forest algorithm [12], decision tree [13], RNN [14]and so on shows less error.

Improving predictive accuracy would be of extreme importance. If the prediction of congestion could be combined with navigation, navigation will provide better outcomings of the shortest road for passengers. For passengers, it can effectively save their time by avoiding congestion. Because some roads may have less distant length, but they are in heavy traffic [15]. For drivers, it can save their time as well, allowing them to take more passengers so that they can increase their income. Moreover, it can contribute a lot to increase the mean speed by preventing traffic jams. Though researchers have used many machine learning methods to predict traffic flow, but few people pay attention to the boundaries of different machine algorithms. In this paper, we use AdaBoost, Random Forest, SVM and BP neural network to predict short-term traffic flow in California. We compare the differences in prediction performance of different algorithms and analyze their potential reasons, which we believe may provide some new insight in the task of short-term traffic prediction.

2. Method

In this section, we first revisit the details of different algorithms, including the Random Forest, AdaBoost, SVM and BP neural network.

2.1. Random Forest

Random Forest is an algorithms which gathering many decision trees via bootstrap aggregating. The steps of bootstrap aggregating are stating as followed:

Firstly, selecting sub-data sets of size n and then put back for k times from a train set of size n. At the same time, obtain k models through training and learning. Secondly, utilize these models to predict and get several results. Thirdly, calculate the mean of these results and gain a final result. Bootstrap aggregating can operatively decrease the chance of overfitting and contribute a lot to control noisy labels.

The process of Random Forest algorithm is: first sample K times randomly and retroactively from a training dataset of size N with M features, and the size of each sub-data set is n. Then, m features are randomly selected for each sub-data set, and a complete decision tree is learned with these n data and m features. Finally, these K decision trees are used to form a Random Forest to obtain the final prediction result. The Random Forest algorithm can process a large number of input variables, assess the importance of variables, automatically estimate lost data, and shorten training time. However, it also sacrifices the interpretability of the decision tree and can still overfit when dealing with some noisy data.

2.2. AdaBoost

AdaBoost is short for Adapt Boost. Its purpose is to form a series of weak classifiers and basic classifiers from the training data and combine them into a strong classifier. Its specific algorithm is shown below in combination with a set of formulas.

Before starting the algorithm, input a set of training data $\mathrm{x_i}$ and corresponding data label $\mathrm{y_i}$. The weight distribution of the training sample set is $\mathrm{D_t(i)}$. A weight size is $\mathrm{w_i}$ for each training sample. A weak classifier is h and a basic classifier is $\mathrm{H_i}$. $\mathrm{H_{fin}}$ means the final strong classifier and $\mathrm{\alpha_t}$ denotes the weight of the weak classifier. An error rate is $\mathrm{e_t}$.

The first step of the algorithm is to initialize the weight distribution of the training data, assigning the same weight to each training sample, that is:

$$D_1(i) = (\frac{1}{N},\frac{1}{N},\dots,\frac{1}{N}) \tag{1}$$

The second step is iteration. T is used to represent the number of iterations, and a total of T times are reproduced. At each iteration, a weak classifier h with the lowest current error rate is selected as the t basic classifier $\mathrm{H_t}$, and a new weak classifier $\mathrm{h_t}$ is calculated, the error of which will be:

$$e_t = \sum_{i=1}^{N} w_{ti} I(H_t(x_i) \neq y_i) \tag{2}$$

The weight in the final classifier will be:

$$\alpha_t = \frac{1}{2} ln\left(\frac{1-e_t}{e_t}\right) \tag{3}$$

The recurrence formula is:

$$D_{t+1}(i) = \frac{D_t(i)e^{-\alpha_t y_i H_t(x_i)}}{2\sqrt{e_t(1-e_t)}} \tag{4}$$

So that:

$$D_{t+1}(i) = \begin{cases} \frac{D_t(i)}{2e_t} & wrong\ sample \\ \frac{D_t(i)}{2(1-e_t)} & right\ sample \end{cases} \tag{5}$$

Iterate until getting the final weight distribution.

The third step is to combine the weak classifier according to the final weight distribution, and obtain the final strong classifier through its symbolic function, that is:

$$H_{fin} = sign(\sum_{t=1}^{T} \alpha_t H_t(x)) \tag{6}$$

The accuracy of AdaBoost is very high and the precise of each classifier is fully considered, but the number of iterations is not easy to determine, and the training for AdaBoost is time-consuming.

2.3. Support vector machines (SVM)

The basic principle of SVM learning is to solve the separation hyperplane able to correctly partition the given dataset with the maximum margin. The problem is often transformed into an optimization problem to minimize $\frac{\|w\|^2}{2}$ to satisfy the restriction, as:

$$y_i(w^T x_i + b) \geq 1 \tag{7}$$

where $\mathrm{y_i}$ is the data label, $\mathrm{x_i}$ is the data vector, w is a certain normal vector of the hyperplane and b is an intercept.

Sometimes some data points deviate from most of the points with the same data label. A slack variable, $\mathrm{\xi_i}$, can be introduced at this point to discard some of the decision weights for these points. With n data points, the problem changes to find the minimum value of $\frac{\|w\|^2}{2} + \mathrm{C}\sum_{i=1}^{n} \xi_i$ to satisfy

$$\begin{cases} y_i(w^T x_i + b) \geq 1 - \xi_i \\ \xi_i \geq 0 \end{cases} \quad i = 1,2,\dots,n \tag{8}$$

The C here is called the penalty factor. The larger the penalty factor is, the less tolerable errors are. In other cases, data points cannot be separated linearly at all, and we usually convert every low-dimensional data vector x_i into a high-dimensional data vector $\varphi(x_i)$ to disperse the linearly inseparable data into new dimensions, and replace every x_i in the optimization problem by $\varphi(x_i)$. Usually, SVM doesn't need to get the explicit expression of $\varphi(x_i)$ and only needs a kernel function, as:

$$K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j) \tag{9}$$

To solve the optimization problem after the replacement, the strong duality theorem is applied according to the optimization theory, and the problem is transformed into a duality problem, that is, to maximize

$$\theta(\alpha, \beta) = \min_{all\ (w, \xi_i^{'}, b)} \left\{ \frac{\|w\|^2}{2} - C \sum_{i=1}^{n} \xi_i^{'} + \sum_{i=1}^{n} \beta_i \xi_i^{'} + \sum_{i=1}^{n} \alpha_i \left[1 + \xi_i^{'} - y_i w^T \varphi(x_i) - y_i b\right] \right\} \tag{10}$$

so that,

$$\begin{cases} \alpha_i \geq 0 \\ \beta_i \geq 0 \end{cases} \quad i = 1, 2, \ldots, n \tag{11}$$

Here, $\xi_i' = -\xi_i$. This is a convex optimization problem, finally being transformed into the problem to maximize

$$\theta(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{12}$$

so that,

$$\begin{cases} 0 \leq \alpha_i \leq C \\ \sum_{i=1}^{n} \alpha_i y_i \geq 0 \end{cases} \quad i = 1, 2, \ldots, n \tag{13}$$

To train SVM, input training samples, solve the optimization problem above, and solve b with the help of KKT condition, then the training can be completed. To test, enter test data, take one piece of test data x, and if for the x and the training dataset x_i ($x_i = 1, 2, \ldots, n$), the formula

$$Y = \sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b \tag{14}$$

is positive (or negative), specify a label of +1 (or -1) for this test data.

2.4. Back propagation neural network

Backpropagation networks are trained with the generalized delta learning rule. It can be divided into three layers: the input layer, the hidden layer and the output layer. Every layer contains many notes. Here we adopted Sum of the Squared Error as the loss function. Based on Backpropagation algorithm, we can obtain 4 formulas:

$$\delta_i^{n_l} = -\left(y_i - a_i^{n_l}\right) \cdot f'(z_i^{n_l}) \tag{15}$$

$$\delta_i^l = \sum_{j=1}^{S_{l+1}} \left[\delta_j^{l+1} \cdot w_{ji}^l\right] f'\left(z_i^l\right) \tag{16}$$

$$\frac{\partial}{\partial_{w_{ij}}^l} J(w, b) = a_j^l \delta_i^{l+1} \tag{17}$$

$$\frac{\partial}{\partial_{b_i}^l} J(w, b) = \delta_i^{l+1} \tag{18}$$

Where n_l means the number of the network layers. l means the layer l, for example, if $l = 5$, that means the fifth layer. Note i means the certain note, for instance, Note 5 means the fifth note.w is for weight, so w_{ij}^l means the weight between the Note i in Layer $l + 1$ and Note j in Layer l. b is for bias, so b_i^j means the bias of Note i in Layer l. z_i^l is the total weight of Note i in Layer l. a_j^l means the active value (output value) of Note i in Layer l. S_l is the number of notes in Layer l. y_i means the practical value.

2.5. Parameter settings

There are details of our parameters. We chose 0.082 seconds as the training time for the Random Forest. The data were split into 0.8 pieces. The internal node splitting sample minimum was two. The maximum depth for a tree was set at 50, and that was also the maximum depth. There were 100 decision trees, and we used a sample-back strategy.

We chose 0.161 seconds as the training time for Adaboost. The data were split into 0.8 pieces. There were 100 base classifiers. The rate of learning was 0.4. A decision tree classifier served as the foundation classifier. We chose 0.007 seconds as the training time for SVM. The data were split into 0.8 pieces. The fine multiplier was 1. Three terms accounted for the most kernel functions. The requirement for error convergence was 0.001. Moreover, a maximum of 1000 iterations was allowed. The initial learning rate for the BP Neural Network was set to 0.01. The rate of learning was 'invscaling'. Inverse scaling learning rate with an exponent of 0.2. 'Adam' served as the weight optimization solution. 'Tanh' served as the hidden layer's activation function. The L2 regularization term's strength was 0.1.

3. Data analysis

3.1. Merits

To begin with, the short-term traffic prediction is a hot issue these days, particularly the prediction for the main lane. Table 1 recorded the traffic state on a main lane of California, the United States, which is typical and can be applied to other similar circumstances. It recorded the traffic flow and traffic speed between March 1st and March 7th (both included) with five-minute time interval.

Secondly, this data set is detailed and explicit enough to ensure the accuracy of prediction. Numerous data sets are available on the website now, but almost all of them simply elaborate the current situation and offer some data which has already been processed.

Thirdly, the most precious merit is its time interval. The five-minute time interval is rare among those data sets. Taking the data set provided by the local government of Shenzhen (Table 2.) as an example, the time interval is 24 hours, which is relevantly accurate to some extent. Because the majority of available data sets are not intact or have a longer time interval. With the aim of predicting short time traffic flow, five-minute interval is apparently more appropriate than 24 hours.

Then, the record time begins in March 1st and end in March 7th, avoiding popular festival which may make a big difference to the traffic. Compared with those data sets which lasting for a long period of time, this is more accurate for prediction in ordinary days.

Table 1. Five-minutes time interval.

Time	Speed	Flow
3/01/2019 0:00	69.40	48
3/01/2019 0:05	69.40	47
3/01/2019 0:10	69.20	49
3/01/2019 0:15	69.00	48
3/01/2019 0:20	68.80	44
3/01/2019 0:25	68.60	41
3/01/2019 0:30	68.60	40
3/01/2019 0:35	68.60	39
3/01/2019 0:40	68.10	38
…	…	…

Table 2. Every 24 hours' traffic flow.

REC_TIME	AVGFLOW	AVGSPEED
25-6 12:00:00.	17161	41.4
26-6 12:00:00.	10972	53.3
27-6 12:00:00.	3014	56.5
28-6 12:00:00.	8759	59
29-6 12:00:00.	12818	38.2
30-6 12:00:00.	22908	31
01-7 12:00:00.	11236	36.2
02-7 12:00:00.	13422	31.4
03-7 12:00:00.	0	0
…	…	…

3.2. Preprocess

Firstly, time series, the X variable, need to be altered because it is data format which is not compatible for most of the mathematic modeling methods. It is supposed to be converted to number format to ensure the X variable is valid.

Secondly, min-max normalization is indispensable. Because when the value of the data set is too large, even if it is multiplied by a small weight, it is still a large number, and the output of the activation function tends to 1 in the activation function, which is not conducive to learning. Here is the formula of min-max normalization:

$$x^* = \frac{x-min}{max-min} \tag{19}$$

Then it can be utilized in some mathematic modelling methods.

4. Result analysis and the improvement of accuracy

4.1. AdaBoost

We utilized these machine learning methods which we mentioned before in SPSSPRO to make a preliminary result analysis. As shown in Table 3, the results made via AdaBoost and Random Forest are impressive because they are fitting fairly well. Particularly in AdaBoost method, in train set R^2 was 1 and R^2 in test set was extremely close to 1, which means that it was not over-fitting.

Table 3. Results made by AdaBoost methods.

	MSE	RMSE	MAE	MAPE	R^2
Train set	0	0.002	0.001	0.589	1
Test set	0	0.02	0.013	5.361	0.996

Then, we used 80% of the data for training and did pre-disruption. We set the learning rate to 1, applied 100 decision trees as base classifiers, and set the loss function to be linear. The R^2 of test set was also close to 1, and that of the training set was shown exactly as 1. The results are shown in Table 4 and Figure 1, also taking March 4 as an example.

Table 4. The model evaluation results for AdaBoost.

	MSE	RMSE	MAE	MAPE	R^2
Train set	0	0.003	0.001	1.649	1
Test set	0	0.017	0.013	6.259	0.997

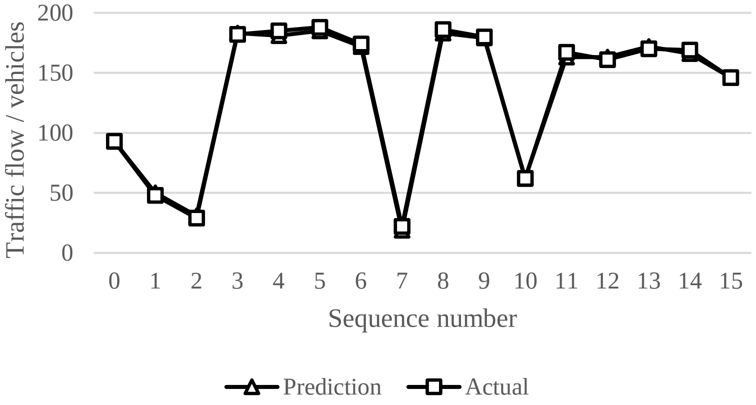

Figure 1. Predicted value obtained using AdaBoost (part).

It needs to be named that the MSEs for both methods were exactly 0, also proving their effectiveness. For Adaboost algorithm, this paper adjusted the learning rate, the types of base learners and the number of base learners. No significant change in model accuracy was observed because the project is to predict short-term passenger flow.

4.2. Random Forest

80% of the data was used for training and pre-disrupted the data. MSE was used to evaluate node splitting, the minimum sample number of internal node splitting was set to 2, and the minimum sample number of leaf nodes was set to 1. We gathered 100 decision trees with the maximum tree depth of 10 and maximum number of leaf nodes of 50. This paper split the data by date, and observation found that the R^2 of both the training set and the test set were extremely close to 1, indicating perfect effects. The results are shown in Table 5 and Figure 2, taking March 4 as an example.

Table 5. The model evaluation results for Random Forest.

	MSE	RMSE	MAE	MAPE	R²
Train set	0	0.008	0.006	3.489	0.999
Test set	0	0.017	0.013	3.643	0.997

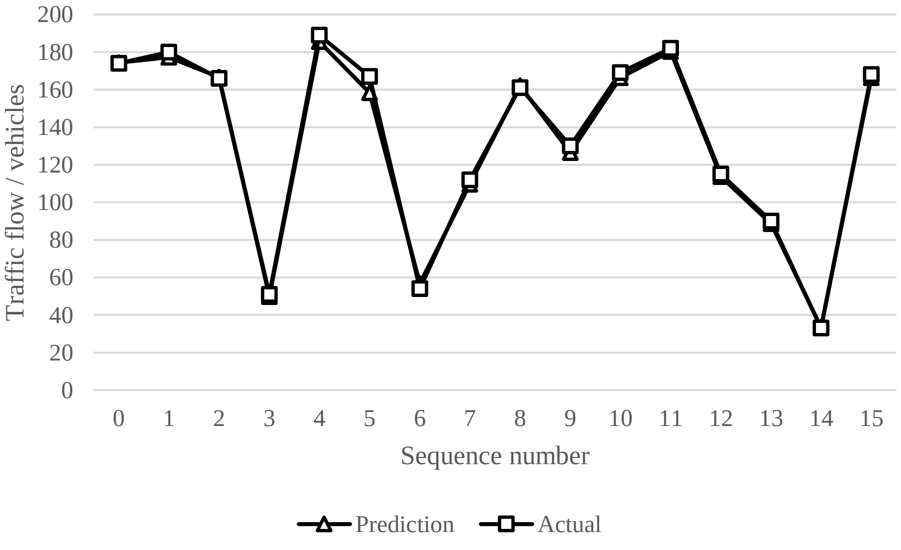

Figure 2. Predicted value obtained using Random Forest (part).

For Random Forest algorithm, this paper first adjusted the number of decision trees and observed no significant change in model accuracy. Then the maximum depth of the tree was adjusted and it was observed that as the value increased, the model gradually became inaccurate. When the maximum depth was 2, the R^2 was extremely close to 1; with the maximum depth of 64, the R^2 had fallen between 0.96 and 0.97. The reason was that increasing the maximum depth of a tree may make the model too complex which is opposite to accuracy.

4.3. *Support vector machines (SVM)*

With regard to SVM, however, the results were not as good as that in Random Forest and AdaBoost. As shown in Table 5, although it showed a high fitting precision, R^2 decreased to approximately 0.8. So there must exist something that can be altered to improve the results via this method.

Table 6. Results made by SVM methods.

	MSE	RMSE	MAE	MAPE	R^2
Train set	0.022	0.148	0.119	31.175	0.799
Test set	0.028	0.167	0.13	29.576	0.758

However, we found that if we use one day's data (See in Table 6 and Figure 3), the results would be much more precise than that showed in preliminary analysis.

Table 7. Results utilizing data in one day (SVM).

	MSE	RMSE	MAE	MAPE	R^2
Train set	0.006	0.078	0.068	91.498	0.947
Test set	0.007	0.081	0.073	64.684	0.938

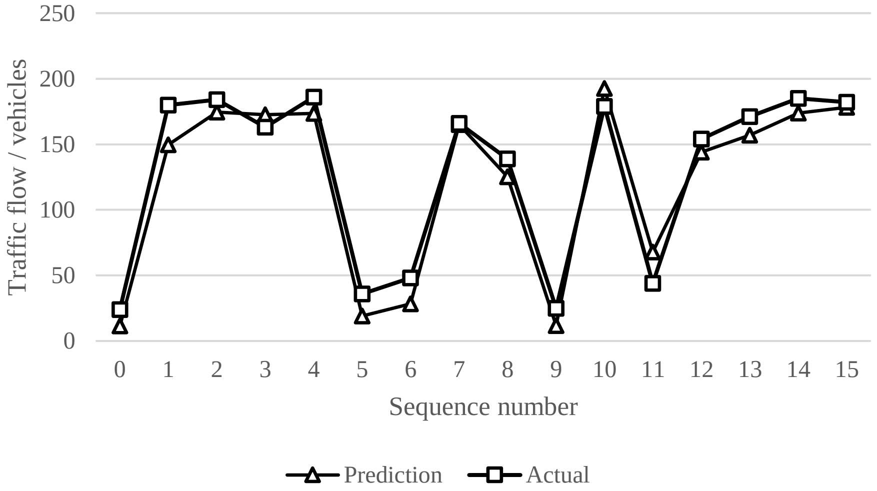

Figure 3. Predicted value obtained from separated analysis (part).

For SVM algorithm, this paper also changed the kernel function coefficient from scale to auto, resulting in a training set R^2 of 0.849 and a test set R^2 of 0.827, indicating less accuracy. Subsequently, the kernel function was changed to linear, and no fit was found, indicating that the data was linearly inseparable.

5. BP neural network

The BP neural network seems inappropriate to make the short-term prediction, or some parameters may need to be adjusted. As shown in Table 7, the results witness a pretty low fitting precision.

Table 8. Results made by BP neural network methods.

	MSE	RMSE	MAE	MAPE
Train set	0.109	0.33	0.289	44.96
Test set	0.104	0.322	0.283	44.679

We found that it was due to one characteristic of the data set that made a huge difference between the two results, the periodicity of the dataset. Some algorithm are creating to specialize the prediction of long-term traffic flow, which can recognize the periodicity of the dataset. Thus, they will not regard the dataset as the ordinary dataset. Nevertheless, the Neural Network is a versatile algorithm, not specializing in handling one kind of problem, which means it cannot recognize the periodicity. Hence, it treated the dataset as the ordinary one. So we divided the data set into 7 parts, each part denotes the traffic flow of 5 minutes time interval in one day. In this case, as shown in Table 8, the predict value was much better than the previous.

Table 9. Results utilizing data in one day (BP Neural Network).

	MSE	RMSE	MAE	MAPE	R2
Train set	1343.799	38.733	32.955	0.413	0.845
Test set	1500.277	36.658	35.566	0.391	0.877

The parameters of BP Neural Network are what we should not ignore. Unlike other algorithms, BP Neural Network has relevantly more parameters. Also, the range of some parameters was extensive. So in order to find the best parameters, we used grid search in python to find them. Finally, we found that the best parameters were:

(1) The initial learning rate used: 0.01;
(2) Learning rate: invscaling;
(3) The exponent for inverse scaling learning rate: 0.2;
(4) The solver for weight optimization: adam;
(5) Activation function for the hidden layer: tanh;
(6) Strength of the L2 regularization term: 0.1;
(7) Hidden layer sizes: 100.

After reset these parameters, the results were much better, as shown in Table 9 and Figure 4.

Table 10. Results utilizing data in one day (BP Neural Network with parameters improved).

	MSE	RMSE	MAE	MAPE	R^2
Trainset	537.543	23.185	16.972	0.188	0.934
Test set	686.599	26.203	19.095	0.200	0.941

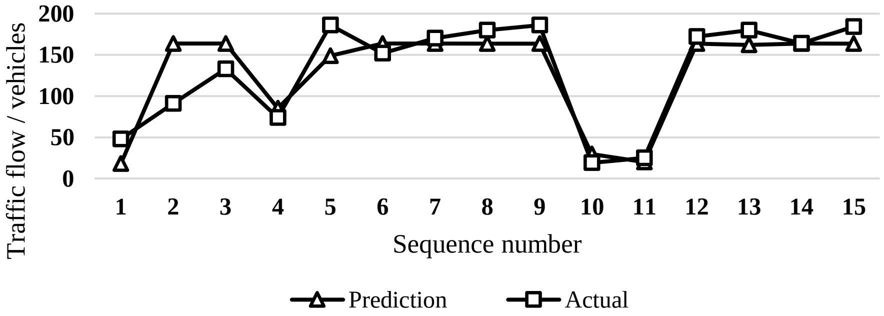

Figure 4. Predicted value obtained from separated analysis (part).

If the hidden layer sizes increased to 200 or larger, gradient disappearance seemed to come out and the R2 began to decrease. When the learning rate was changed, the result nearly remained. That was because the model was not too complicated. When choosing 'Relu' or 'identity' as the activation function, the result was much worse than selecting 'tanh' due to the fact that the short-term traffic model is the non-linear model, so employing 'tanh' or 'logistic' would be much better.

6. Conclusion

The Random Forest and AdaBoost showed high precise in predicting short-term traffic flow, no matter how big the volume of the data set is. Because these two algorithms can recognize the periodicity of the dataset. In this case, they can fit better than SVM and BP neural network. The latter two algorithms cannot discern the periodicity, so that they can only handle with data within one cycle. Thus, when we are going to employ these methods, picking out data within one cycle is indispensable, or it may be more appropriate to utilize them when the dataset takes more factors into consideration and constrains the place and time as well. Meanwhile, unlike the SVM, BP neural network has more parameters, so adopting them will play an important role in improving the fitting precise. In addition, despite that the parameters of BP neural network were modified and the R^2 was close to 1, the MSE was too high. Hence, the optimization algorithm of BP neural network might need alteration for the gradient descent that BP neural network employed might lead to locally optimal solution.

References

[1] Lippi M., Bertini M., 2013. J. Frasconi P..Short-term Traffic Flow Forecasting: an Experimental Comparison of Time-series Analysis and Supervised Learning. IEEE Transactions on Intelligent Transportation Systems,14(2).

[2] D. Schrank, B. Eisele, T. Lomax. 2012. J. Urban Mobility Report Powered by INRIX Traffic Data.

[3] Doğan, E. 2022. J. Robust-LSTM: a novel approach to short-traffic flow prediction based on signal decomposition. Soft Computing, 26(11).

[4] Zhao H., Zhai DM., Shi CH. 2019. J. A Review of Short-Term Traffic Flow Prediction Models. Urban Rapid Transit, 32(04).

[5] Ye, Y., Chen, L., Xue, F. 2019. J. Passenger Flow Prediction in Bus Transportation System Using ARIMA Models with Big Data. International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery.

[6] Okutani I., Stephanedes Y.J. 1984. J. Dynamic prediction of traffic volume through Kalman filtering theory. Transportation Research Part B ,18 (1), pp. 1-11.

[7] Liao, RH., Lan, SY., Liu, ZX. 2015. J. Short-term Traffic Flow Forecasting Based on Local Prediction Method in Chaotic Time Series, Computer Technology and Development.

[8] Forbes G.J., Hall F.L. 1990. J. Applicability of Catastrophe Theory in Modeling Freeway Traffic Operations. Transportation research Part A: general , 24A (5) :335-344.

[9] He, GG., Ma, SF., Li, Y. 2017. J. Study on the Short-term Forecasting for Traffic Flow Based on Wavelet Analysis. Systems Engineering-theory and practice, (2002), 22 (9) :101-106.

[10] Cheng, SY. 2017. J. Short-term Traffic Flow Prediction Method Based on Fuzzy Neural Network Research. Computer measurement and control , 25 (8) :155-158.

[11] Yan, KW. 2009. J. Study on the Forecast of Air Passenger Flow Based on SVM Regression Algorithm. 1st International Workshop on Database Technology and Applications.

[12] Han C., Ma T., Xu G., Chen S., Huang R. 2022. J. Intelligent decision model of road maintenance based on improved weight Random Forest algorithm. International Journal of Pavement Engineering,23 (4), pp. 985-997.

[13] Fu, X., Wang, D., Zhang, HC. 2018. J. Identifying Transportation Modes Using Gradient Boosting Decision Tree. ICIC.

[14] Zou, ZN. 2018. J. Deep Convolutional Mesh RNN for Urban Traffic Passenger Flows Prediction. SmartWorld.

[15] Lippi M., Bertini M., Frasconi P. 2013. J. Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning. IEEE Transactions on Intelligent Transportation Systems, 14 (2), art. no. 6482260, pp. 871-882.