Research Article

Temporal Backtracking and Multistep Delay of Traffic Speed Series Prediction

Licheng Qu,1,2 Minghao Zhang,2 Zhaolu Li,2 and Wei Li1

1School of Computer Science and Technology, Xi’an Jiaotong University, Xi’an 710049, China
2School of Information Engineering, Chang’an University, Xi’an 710064, China

Correspondence should be addressed to Wei Li; liw@xjtu.edu.cn

Received 28 September 2020; Revised 20 November 2020; Accepted 30 November 2020; Published 11 December 2020

Abstract

As a typical time series, the length of the data sequence is critical to the accuracy of traffic state prediction. In order to fully explore the causality between traffic data, this study established a temporal backtracking and multistep delay model based on recurrent neural networks (RNNs) to learn and extract the long- and short-term dependencies of the traffic state data. With a real traffic data set, the coordinate descent algorithm was employed to search and determine the optimal backtracking length of traffic sequence, and multistep delay predictions were performed to demonstrate the relationship between delay steps and prediction accuracies. Besides, the performances were compared between three variants of RNNs (LSTM, GRU, and BiLSTM) and 6 frequently used models, which are decision tree (DT), support vector machine (SVM), k-nearest neighbour (KNN), random forest (RF), gradient boosting decision tree (GBDT), and stacked autoencoder (SAE). The prediction results of 10 consecutive delay steps suggest that the accuracies of RNNs are far superior to those of other models because of the more powerful and accurate pattern representing ability in time series. It is also proved that RNNs can learn and mine longer time dependencies.

1. Introduction

As an indispensable means of transportation, automobiles have brought unprecedented convenience to our daily life. At the same time, they have also brought many social problems, such as traffic congestion, energy crisis, and air pollution, which have caused widespread concern around the world [1, 2]. Modern intelligent transportation systems (ITS) are the practical (perhaps the only) way to solve these increasingly serious problems [3]. Among them, advanced traveller information systems (ATIS) and advanced traffic management systems (ATMS) are playing a huge role in updating route information, forecasting future traffic conditions, reducing traffic congestion, and improving the overall efficiency of the transportation network [4, 5]. As the cornerstone of intelligent transportation systems, traffic flow forecasting is crucial to the development of ITS. Accurate and reliable traffic state prediction can help road users and managers to grasp the real-time traffic status and plan travel routes more reasonably. Objectively, it also reduces environmental pollution and improves the management art of smart cities [6].

Due to the subjectivity and stochastic characteristics of travel demand, the state of traffic flow shows different patterns under the influence of various factors [7]. It is not easy to accurately predict future traffic conditions. In recent years, with the rapid spread of traffic sensors, real-time traffic data have been continuously collected and stored. Traffic data plays an important role in assessing traffic conditions and in the application of urban traffic intelligence management [8]. Abundant historical data make it possible to accurately predict future traffic patterns, which have also led to a large number of data-driven traffic prediction studies.

At the same time, traffic flow has a strong time correlation. It repeats the same cycle every day, from the low traffic in the morning to the high traffic during working hours, and finally back to the low traffic in the evening. This periodic feature makes the traffic states data a typical time series, and each data in the series is related to time. Time and sequence characteristics are important attributes that cannot
be ignored in traffic state data. By recognizing and utilizing the correlations between traffic patterns and contextual factors [9], the traffic patterns mined from the data sequence will reflect the future of the traffic state.

Among all the time series forecasting methods, deep learning methods generally show better performance. Particularly, recurrent neural networks (RNNs) have more advantages than other models in time series analysis because they can memorize their internal state to process input sequences [10]. Long short-term memory (LSTM) neural networks, as the most popular variant of RNNs, have received much attention in recent years [11]. However, the potential of deep learning methods in traffic forecasting has not yet fully been exploited [12]. However, the LSTM can automatically determine the optimal time lags [13], to obtain better accuracy, finding that the optimal time lags setting largely relies on the trial-and-error method. Obviously, if the data sequence is too short, it is impossible to fully explore the time relationship between the sequences, which directly leads to a decrease in prediction accuracy. However, too many backtracking steps will bring more calculations, but the prediction accuracy will not be significantly improved. How to strike a balance between sequence length and prediction accuracy is one of the concerns of this research.

Another issue studied in this paper is forecasting across multiple time steps. Almost all current researches focus on the prediction of the next time step, and the delay prediction that skips multiple time steps is very rare. Using data and models of the current time interval to predict traffic conditions after multiple time steps (e.g., 2 hours and 3 days) is of great significance in practical applications. It can help travellers and managers to predict traffic conditions in advance and take countermeasures as early as possible. Similarly, the more the time steps span is, the more the prediction error will increase. Therefore, another focus of this paper is to study the relationship and impact between the number of delay steps and the prediction accuracy to meet the needs of practical applications and provide inspiration for future research.

In order to fully explore the temporal relationship between traffic sequences, an LSTM which has a stronger and more accurate time series pattern learning capability was established to learn and extract the long-term and short-term time dependencies. At the same time, the coordinate descent algorithm was used to search and determine the backtracking length of the sequential data. Finally, the traffic state predictions with a multistep delay were performed to evaluate the correlation between the delay steps and the prediction accuracy.

Based on the aforementioned discussion, the contributions of this paper are 3-fold: (1) a traffic sequence prediction model with multistep delay was established, and the optimal length of the backtracking sequence is determined by the coordinate descent algorithm. (2) The approximately linear relationship between prediction accuracy and the delay steps was evaluated and demonstrated. (3) It was found that RNNs can learn and represent longer temporal dependence and are very suitable for time series causality mining. In addition, this research is also of great significance to actual traffic management. The optimization of the number of backtracking steps can improve the accuracy and reliability of traffic prediction and avoid wasting more computing resources. It can also guarantee the real time and reliability of the forecast. Multistep delay prediction can help management use limited sequential data to perceive long-term road network conditions as early as possible and adjust road control plans in time. Long-term forecasts at multiple time steps issued by the management department can also help travellers grasp the dynamic traffic changes during the entire travel period and help choose more optimized travel routes to avoid congestion. Particularly, during peak hours and emergencies, the ability of providing early warning of traffic conditions across multiple time steps will bring more benefits to traffic management and travel planning.

The remainder of this paper is organized as follows: Section 2 reviews the related works on traffic state prediction. Section 3 introduces the data definitions and methods. Section 4 demonstrates the optimized process of time backtracking and evaluates the performance of multistep delay prediction. Finally, conclusions are drawn with future works in the last section.

2. Literature Review

With the continuous improvement of artificial intelligence theory, machine learning algorithms are widely used in traffic flow prediction, such as decision tree (DT) [14], support vector machine (SVM) [15], k-nearest neighbour (KNN) [16], random forest (RF) [17], and gradient boosting decision tree (GBDT) [18]. Because of the powerful non-linear approximation capabilities, machine learning methods are suitable for short-term traffic flows prediction with strong randomness and can obtain good prediction accuracy. Han et al. demonstrated that machine learning methods (especially SVM and neural networks) can clearly outperform two traditional statistical models [19].

However, due to the shallower architecture of machine learning methods and the lack of deeper representation capabilities, it is difficult for most machine learning predictive models to mine deeper abstract relationships between traffic data. Moreover, many prediction algorithms usually need manual adjustment of parameters. This usually requires experience and skills, which is very time consuming and greatly limits practical applications.

In 2006, Hinton trained a multilayer neural network to reconstruct high-dimensional input vectors and proposed a layered initialization training method to solve the parameter training problem of deep neural networks (DNN) [20]. Since then, a new era of deep learning has started. In essence, deep learning is an expansion of the number of layers of shallow neural networks. The increase in the number of layers gives the deep neural network more nonlinear abstraction and representation capabilities. For example, Huang et al. combined a deep belief network with a multitask regression layer and proposed a deep architecture for traffic flow prediction [21]. Zhou et al. proposed an adaptive boosting scheme which ensembles the trained SAEs to improve the accuracy of traffic flow forecasting [22]. Zhang et al. propose...
a short-term traffic flow prediction model based on a CNN deep learning framework to predict the traffic speed with a spatiotemporal feature matrix [23] and report that the prediction result is better than the support vector machine and the shallow neural network model.

In addition to the deep neural network algorithms used alone, there are also some ensemble learning methods that combine multiple prediction algorithms and much more flexible structure to obtain better predictive performance. Tang et al. propose a new method in construction fuzzy neural network with the k-means method to forecast travel speed for multistep ahead based on 2-minute travel speed data [24]. Ma et al. noticed days which experience significantly different traffic flow patterns, negatively influencing forecasting results, and proposed an advanced method based on a CNN and LSTM model to select the appropriate predictor for pattern matching [25].

Due to the time series characteristics of traffic flow, the RNN model with time series processing as the core has naturally become the most concerned and popular deep model. By introducing gate functions into the cell structure, the LSTM could handle the problem of long-term dependencies very well; since then, almost all the exciting results based on RNNs have been achieved by the LSTM. The LSTM has become the focus of deep learning [26, 27]. Compared with popular time series prediction methods, LSTM can achieve the best prediction performance in accuracy and stability. Tian et al. proposed a learning method based on long short-term Memory, which uses multiscale temporal smoothing employed to infer lost data and the prediction residual [28]. Zhao et al. proposed a traffic prediction model based on long-term and short-term memory networks. It considers temporal–spatial correlation in traffic system via a two-dimensional network which is composed of many memory units. A comparison with other representative forecast models validates that the proposed LSTM network can achieve a better performance [29].

Because of the outstanding performance of LSTM in time series, researchers have paid more attention to and developed many improved models based on it, for example, the gated recurrent unit (GRU) model based on simplified LSTM [30, 31], the convolutional LSTM (ConvLSTM) model for spatiotemporal feature extraction (combining the learning capabilities of convolutional neural networks and recurrent neural networks) [32, 33], and the bidirectional LSTM (BiLSTM) model used to extract the sequential features from forward and backward directions [34, 35].

Deep learning prediction models can mine high-dimensional features of data through semiaffinity nonlinear transformation sequences and learn the nonlinear and nonstationary relations between variables, which have also been found to be suitable for big data analysis with successful applications to computer vision, pattern recognition, speech recognition, natural language processing, and recommendation systems [36]. Deep learning, especially recurrent neural networks, is so successful that more and more ensemble predictive models have been developed based on it. However, few researchers discuss the influence of the number of backtracking steps on the prediction of traffic flow time series and the exploration of multistep delay prediction using existing models. These are of great significance for a better understanding of deep neural networks and time series and are exactly what this paper focuses on.

3. Methodology

In this section, the algorithm framework is introduced first, then the data sequence used is defined, and, finally, the recurrent neural network model and parameter optimization method used are briefly explained.

3.1. The Framework of Traffic Flow Time Series Prediction.

The general framework of the traffic flow time series prediction proposed in this paper is shown in Figure 1. The process of prediction consists of two stages: model training and model prediction. In the model training stage, the sequential data of traffic flow was organized as subsequence and fed into the pattern recognition and extraction module for training. These sequences can be consecutive data or data with a certain interval. In the entire training set and test set, a unified data format is maintained. The training module continuously evaluates and updates the model. Finally, the RNN model is generated, which can recognize and extract the patterns and features of the traffic flow time series.

In the prediction stage, the prepared backtracking sequence is sent to the traffic flow sequence prediction module, then the trained RNN model is loaded and the traffic state prediction is performed with multiple delay steps, and, finally, the prediction result is output.

3.2. Definition of Sequential Data.

Time series is a sequence of numbers in which the values of the same statistical indicator are arranged in the order of their occurrence time. Each value in the time series is the result of successive observations of the same phenomenon at different times and is the result of the joint action of different factors. By statistically analysing the regularity of the collected time series, we can obtain the regularity of data changes and predict future development trends.

In general, traffic state data are collected in a given time interval, such as 5 minutes, 15 minutes, or 1 hour. This time interval is also called the sampling interval. No matter which interval is used, the sampled data will be accompanied by a fixed sampling time. At the same time, for traffic speed, this data represents the average speed of vehicles passing by during the sampling interval. Theoretically, x can be used to represent traffic flow time series data for N days, where $x = (x_1, x_2, \ldots, x_N)$, and $\mathbf{x}$ represents the traffic flow data on the $i$-th day, which is expressed as follows:

$$\mathbf{x} = \{x^i_{t}\}_{t=1}^{T} = (x^i_1, x^i_2, \ldots, x^i_T),$$

where $x^i_t$ is the traffic state data at the $t$-th time point on the $i$-th day, $t \in [1, T]$, and $T$ is the number of data points per day in the current sampling interval, $T \in \mathbb{N}^*$. As we all know, the closer the data is to the current time, the greater the impact on the future model is. Therefore, in
the process of traffic state forecasting, it is necessary to go back to the previous data and build a model to find the relationship between historical data and future state, and then use it to predict the future traffic flow. Obviously, a too-long backtracking sequence will lead to too much calculation and the complexity of the model will increase. However, if the backtracking data are too short, the change trends of the traffic series cannot be fully explored, and then the accuracy of the forecast will be reduced. Therefore, it is necessary to determine how many steps need to be backtracked to obtain accurate forecast results while ensuring that the model has low computational complexity. Assuming that the number of backtracking steps is $L_b$, the data sequence used for the prediction can be written as follows:

$$x_t^i = \left\{ x_{t-L_b+t}^i \right\}_{t=1}^{L_b} = \left( x_{t-L_b+1}^i, x_{t-L_b+2}^i, \ldots, x_t^i \right), \quad (2)$$

where $x_t^i$ is the $i$-th data sequence used for prediction on the $t-$th day; $L_b$ is the number of backtracking steps, and $L_b = 1$ means single step backtracking. It can be seen from the formula that the data sequence used for prediction is a new subsequence composed of $L_b$ data. Since the prediction uses the data before the current time point $t$, the sequence which will be used for future prediction may contain the actual data of the previous day or more. So, the time series prediction of traffic state is to trace back the $L_b$ data before the current time point (here marked as $t$) to form a new subsequence to predict the traffic state at the future time point.

Of course, time series forecasting can predict not only the value that will occur at the next time point, but also the value that spans multiple time steps. In the paper, the number of steps that spanned in the forecast is called the number of delay steps, noted by the symbol $D_s$, where the subscript $s$ represents the number of specific delay steps, $s \in \mathbb{N}^*$. Correspondingly, the real value delayed from the current time point $t$ can be written as follows:

$$y_t^{i+D_s} = x_{t+D_s}^i, \quad (3)$$

where $y_t^i$ represents the true value at the $t$-th time point on the $i$-th day, which is used to calculate the response loss and tune the weights during model training; $D_s$ is the number of delay steps, $D_s = s$ in numerical value; the definitions of other symbols are the same as those above. Now, the traffic flow time series prediction will be redefined as constructing a subsequence with backtracking $L_b$ time steps and forecasting the traffic state after $D_s$ delay steps.

### 3.3 Long Short-Term Memory Neural Network

Long short-term memory (LSTM) is the enhancement and improvement of recurrent neural networks. It was firstly proposed by Hochreiter and Schmidhuber in 1997, to overcome the disappearance and explosion of gradients in the back-propagation stage of recurrent neural networks [37].

LSTM is usually enhanced by the recursive gate called forget gate, replacing the constant error carousel (CEC) weight with the forget gate activation multiplication, so that the internal memory state can be in the information flow; it resets itself when it is out of date [38]. The “selective forgetting” ability can remember the long-term information dependence and ensure that useful information will not be diluted by the latest information gradually with the passage of time. This makes the LSTM network suitable for the classification, processing, and prediction of time series data.

An ordinary LSTM unit is composed of a self-connection memory cell and three control gates related to it, as shown in Figure 2. The self-circulation connection updates the state of the memory cell during the time step cycle without external interference. Input gate $i$ and output gate $o$ can allow or prevent information from entering or leaving the recurrent unit. The forget gate $f$ can adjust the memory cell, allowing it to remember or forget its previous state. Over time, these mechanisms enable LSTM to remember or forget certain information [39].

In Figure 2, the input gate and the forget gate use the current input $x_t$ and the output response $h_{t-1}$ of the previous time step to calculate the opening or closing degree to determine whether to accept the current input or forget the previous memory. It can be calculated as follows:

$$i_t = \sigma(x_tW_i + h_{t-1}W_{hi} + b_i),$$

$$f_t = \sigma(x_tW_f + h_{t-1}W_{hf} + b_f), \quad (4)$$

where $x_t$ is the input vector at time step $t$; $h_{t-1}$ is the output response of the previous time step; $W$ and $b$ are the weight matrix and bias vector, which need to be determined during the training process; $\sigma$ is the activation function, which is usually a sigmoid function in practice. In Figure 2, $C_t$ is the cell state of the time step, which allows information to flow along the network without disappearing or exploding. In each time step, the current cell state $C_t$ is generated by adding the previous state $C_{t-1}$ adjusted by the forget gate and the activation input $g_t$ of the current time step. The updated formula is defined as follows:
In LSTM, \( h_{t-1} \) will also be fed back to the input to recursively update the next internal memory state. The output gate uses the current input \( x_t \) and the previous output response \( h_{t-1} \) to calculate the degree of opening or closing, to determine how much newly generated memory information can be output. The output response \( h_t \) at each time step is adjusted by the output gate \( o_t \), and the formula is as follows:

\[
\begin{align*}
    o_t &= h(x, W_{xo} + h_{t-1} W_{ho} + b_o), \\
    h_t &= o_t \circ h(C_t) 
\end{align*}
\]

where \( h \) is the output activation function, which is usually a hyperbolic tangent function; \( C_t \) is cell state and the initial value \( C_0 \) is set to 0; other symbols are the same as above. After the sequential data is fed to the model, the LSTM unit begins to process it step by step, and each time step outputs an internal memory state, and the response output \( h_t \) of the last step is the expected output of the whole recurrent networks.

**Figure 2:** Internal structure of the LSTM unit.

\[
g_t = g(x, W_{xc} + h_{t-1} W_{hc} + b_c), \\
C_t = f_t \circ C_{t-1} + i_t \circ g_t, \\
g_t = f_t \circ C_{t-1} + i_t \circ g_t, \\
\]

where \( \circ \) represents the Hadamard product or element-wise product, which multiplies the elements at the corresponding positions to obtain a new matrix of the same size; \( g \) is the input activation function, which is usually the hyperbolic tangent function; \( C_t \) is cell state and the initial value \( C_0 \) is set to 0; the rest of the symbol definitions are the same as those above.

In LSTM, \( h_{t-1} \) will also be fed back to the input to recursively update the next internal memory state. The output gate uses the current input \( x_t \) and the previous output response \( h_{t-1} \) to calculate the degree of opening or closing, to determine how much newly generated memory information can be output. The output response \( h_t \) at each time step is adjusted by the output gate \( o_t \), and the formula is as follows:

\[
\begin{align*}
    o_t &= h(x, W_{xo} + h_{t-1} W_{ho} + b_o), \\
    h_t &= o_t \circ h(C_t) 
\end{align*}
\]

where \( h \) is the output activation function, which is usually a hyperbolic tangent function; the initial value \( h_0 \) is set to 0; other symbols are the same as above. After the sequential data is fed to the model, the LSTM unit begins to process it step by step, and each time step outputs an internal memory state, and the response output \( h_t \) of the last step is the expected output of the whole recurrent networks.

### 3.4. Optimization of the Backtracking Step

In the case of limited computing power and resources, the efficiency and performance of the model are particularly critical. Researchers always hope to search for the effective hyperparameters of the model through different optimization algorithms to find the best configuration combination in order to get the best score on the key indicators of the test data set. There are many algorithms for searching and optimizing hyperparameters, which are widely used in various researches. Zhang et al. used a constrained hybrid genetic algorithm (GA) to determine the key parameters of the tire friction coefficient identification magic formula [40]. Xu et al. used a step-wise regression model to determine the coefficients in the study of evaluating bicycle lane safety [41]. Wei et al. used dual coordinate descent to optimize the parameters of support vector machines to minimize the loss on training data [42].

Coordinate descent is a nongradient optimization algorithm. In each iteration of the algorithm, a one-dimensional search is performed along a coordinate direction at the current point to find the local minimum of a function. In the whole process, different coordinate directions are used cyclically. For inseparable functions, the algorithm may not be able to find the optimal solution in a small number of iteration steps. To speed up the convergence, an appropriate coordinate system can be adopted; for example, a new coordinate system in which the coordinates are not related to each other as much as possible is obtained through principal component analysis. Besides, various optimization algorithms have been widely used in practical research. The coordinate descent method is used in the calculation of the LASSO regression coefficient [43] and the optimization algorithm SMO of SVM dual problem [44]. In addition, this method also has a good performance as a tuning algorithm in other time series forecasting researches. Amir Mahdi et al. used the coordinate descent method to determine the parameters in the link prediction of the multiplex network [45].

Considering the large amount of data in the data set, it usually takes a long time to complete a search. When the number of parameters continues to increase, the time consumed by the model will increase exponentially. Without the guarantee of supercomputing capabilities, it is almost impossible to complete the search process. Therefore, this paper adopts the gradual and greedy coordinate descent algorithm, trying to quickly complete the parameter determination with limited computing power and spend the least time.

### 4. Experiment

#### 4.1. Data Description

The proposed model was validated by real-world traffic data from DRIVE Net (Digital Roadway Interactive Visualization and Evaluation Network, http://www.uwdrive.net). The data was collected by Washington State Department of Transportation (WSDOT) and maintained by STAR Lab (Smart Transportation Applications and Research Laboratory). The actual locations of 16 detectors within 160 to 170 miles on the I-5 expressway are shown in Figure 3. For convenience, we used the milepost of the detector to indicate its name (e.g., detector 16,395 represents the detector located at 163.95 miles). The minimum sampling interval of the captured data is 5 minutes. After
processing, historical data was aggregated into different intervals, such as 10 minutes, 15 minutes, and 1 hour. The data came from January 1, 2015, to March 31, 2016. All data was divided into four parts. The data of 2015 was used as the training set, of which 20% was randomly selected as the validation set, the data of the first 2 months of 2016 was used to optimize the parameters, and the data from March 2016 was used as the test set.

In order to better evaluate the performance of the proposed model and facilitate the comparison of different models, this study used mean absolute error (MAE), mean absolute percentage error (MAPE), and root mean square error (RMSE) to evaluate prediction results; the formulas are as follows:

\[
\text{MAE} = \frac{1}{n} \sum_{t=1}^{n} |\hat{y}_t - y_t|,
\]

\[
\text{MAPE} = \frac{1}{n} \sum_{t=1}^{n} \left| \frac{\hat{y}_t - y_t}{y_t} \right| \times 100%,
\]

\[
\text{RMSE} = \sqrt{\frac{1}{n} \sum_{t=1}^{n} (\hat{y}_t - y_t)^2},
\]

where \(y_t\) is the true value of traffic data in the \(t\)-th time point; \(\hat{y}_t\) is the predicted value at the same time point; and \(n\) is the number of samples.

4.2. Determination of the Number of Backtracking Steps

As mentioned in the previous data definition section, the data sequence used for prediction is a new sequence composed of the \(L_b\) past data, where \(L_b\) is the number of backtracking steps. When \(L_b\) is big, the dependence between the sequences contained in the data will be enhanced. Of course, the amount of calculation and the time consumption will also increase exponentially. While \(L_b\) is small, the predicted sequence becomes shorter, and the causal relationship between the sequences is incomplete, which makes it difficult to extract and results in poor accuracy. As a hyperparameter, it is necessary to search and determine the number of backtracking steps based on the distribution characteristics of the actual data to strike a balance between model accuracy and computational cost.

The first parameter in the model that needs to be determined is the number of backtracking steps. Before optimization, other model parameters need to be fixed, and model pretraining and evaluation will be carried out only within the range of backtracking steps. If the evaluation result is developing in a good direction, then it continues to follow the search direction. If the evaluation result does not change or becomes worse, the training will be stopped immediately. Then, the parameter value will be determined according to the previous result, and the search process will end. In the process of coordinated descent, only one parameter is determined at a time, and every iterative evaluation is a one-dimensional search, so the final result is probably not the global optimal solution.

Based on the 5-minute sampling data of the detector 16,395, the step-by-step greedy coordinates descent method is used to search and evaluate the number of backtracking steps of the LSTM model. The number of backtracking steps starts from 2 steps and gradually increases to 20 steps. The sorted data sequence is sent to the model for training. After the training is completed, three error metrics including MAPE, MAE, and RMSE were used to evaluate the prediction effect of the model. The results are shown in Table 1.

In Table 1, the maximum and minimum values of prediction errors are marked in bold. It shows that the maximum errors appear in the minimum backtracking steps (2nd step) and the minimum errors appear in the 14th backtracking step. It can be seen that as the backtracking steps increase, the prediction errors decrease significantly. The visualization of the evaluation results in the table is shown in Figure 4. The \(x\)-axis in the figure is the number of backtracking steps, arranged horizontally according to the evaluation step. Through the analysis of the evaluation data, the following conclusions can be drawn: firstly, the number of backtracking steps has a great influence on the accuracy of the model. As shown in Figure 4, the number of backtracking steps on the \(x\)-axis increases from 2 to 20 steps. At the same time, the errors of the LSTM model measured by the three metrics gradually decrease from its maximum.

![Figure 3: The locations of the 16 detectors used in the case study.](image-url)
value, and the maximum decrease rates are 2.56%, 3.18%, and 3.32%, respectively. This also shows that time series data does have long-term relevance and can be mined through time series models. Secondly, the trends of these three errors are essentially the same at the beginning of the search, and the decline rate is also relatively fast. The inflected point of the smooth decline appeared almost simultaneously in step 14, and the errors also reached the local low. Since then, the fluctuations of the errors have stabilized, and sometimes there has been a slight increase. This phenomenon truly reflects the gradual decrease in prediction error as the backtracking sequence becomes longer. However, too long a sequence not only fails to improve the accuracy of the prediction, but also brings negative impact.

To confirm the generality of the conclusion, another experiment was conducted with a different model (GRU), different data (from detector 16,885), and different sampling intervals (20 min, 30 min, and 60 min). The MAE curves of the predicted results are shown in Figure 5. Except that the lowest points are slightly different and the curves drop more smoothly, comparative experiments have obtained almost consistent results. The detector 16,885 is almost 5 miles away from the first detector used before and has different traffic patterns due to different travel needs, so the evaluation results are representative and convincing.

The choice of the number of backtracking steps is related to the models and data. Different models or data may produce different results, but it is certain that there must be a reasonable range of values. Although the difference in model or data set makes the number of backtracking steps fluctuate randomly within this range, this range can be approached step by step by searching multiple models.

In order to verify whether the same decline trend can be obtained on other models, the 5-minute sampling data of detector 16,885 was used to search the backtracking steps of several frequently used models. The results of the backtracking steps searching are shown in Figure 6. The first 5 models are the typical machine learning model, the 6th model is representative of deep neural network models, and the remaining models are three variants of RNNs discussed in this paper. Since each model has a different sensitivity to the correlation of time series, the structure and characteristics of them are also very different. In order to clearly show the differences between all models, the MAE of each model is shown in Figure 6. Although the prediction accuracy of the model is different, the development trend of backtracking steps is approximately the same which can be roughly divided into three situations. The first is that the errors quickly drop to a local low point and then gradually rise after flattening, such as (a) DT and (c) KNN. The second is that the errors decrease rapidly in the first few steps and then remain stable, such as (b) SVM, (d) RF, and (e) GBDT. The last is that the errors continue to decrease after rapidly falling to a local low point and then fluctuate slightly, such as (f) SAE, (g) LSTM, (h) GRU, and (i) BiLSTM. It can be seen from the trends of the error curves that there may be the most suitable number of backtracking steps, but the dramatic increase in computational cost makes it no longer worthwhile to continue searching.

Although the MAE curves of different models are slightly different, the overall trend is nearly the same. Of course, the number of backtracking steps is not as long as possible. Most models have a maximum limit; too long time series will cause the prediction error to rise instead of falling. To select a

| Errors | Backtracking steps $L_b$ |
|--------|--------------------------|
|        | 2  | 4  | 6  | 8  | 10 | 12 | 14 | 16 | 18 | 20 |  |
| MAPE   | 7.42 | 7.33 | 7.32 | 7.28 | 7.32 | 7.25 | 7.23 | 7.27 | 7.26 | 7.25 |
| MAE    | 2.83 | 2.78 | 2.78 | 2.77 | 2.78 | 2.76 | 2.74 | 2.75 | 2.75 | 2.76 |
| RMSE   | 3.91 | 3.85 | 3.84 | 3.82 | 3.84 | 3.8 | 3.78 | 3.79 | 3.8 | 3.81 |

Figure 4: Prediction errors of the LSTM model with different backtracking steps on detector 16,395. (a) MAPE. (b) MAE. (c) RMSE.
Figure 5: MAEs of the GRU model with different backtracking steps on detector 16,885. (a) 20-minute interval. (b) 30-minute interval. (c) 60-minute interval.

Figure 6: Continued.
reasonable number of backtracking steps between the various models, considering that the random fluctuations increase after the performance index tends to flatten during the decline process, a range can be determined based on the first inflection point value. According to the results shown in figures, considering the versatility of the model, the value range can be set between 6 and 18. Since the dependence of the time series contained in the data set also has a significant impact on the determination of the number of backtracking steps, the number of backtracking steps on different data sets may change significantly. The characteristics and distribution rules of traffic data in different data sets are quite different. For data sets with flat changes, the time dependence that can be used will be more, and the corresponding number of backtracking steps will be longer. On the contrary, for data sets with drastic changes, the correlation of time series will be greatly reduced, and the number of backtracking steps that can be used will be correspondingly shorter. In practical applications, the number of backtracking steps for each data set can be determined by experimental evaluation.

Considering the universality of the model, the fairness of the comparison results, and the complexity of calculation time and space, all the time series models in this paper use the middle value 12 as the number of backtracking steps. It represents exactly one hour when a 5-minute interval is used. This uniform number of backtracking steps can provide a relatively fair benchmark for comparison between models.

4.3. Multistep Delay Prediction of Time Series. According to the data definitions in (1) and (2), the traffic flow time series prediction model can predict the time series skipping multiple time steps. Of course, the greater the number of delayed steps, the greater the randomness of the prediction, and the greater the deviation between the predicted result and the true value. However, if the deviation is within an acceptable range, the delay prediction is meaningful.

Taking the data of detector 16,885 for 5-minute samples, the number of backtracking step $L_b$ is set to 12, and the number of delay step $D_s$ is gradually increased from 1 to 10. A double-layer LSTM model is established to predict the traffic speed delayed from 5 minutes to 50 minutes after the current time point. During the evaluation process, the sampling interval and backtracking steps of traffic data remained unchanged, but delay steps gradually increased. After the model training was completed, the prediction performances of different delay steps were evaluated with three error metrics including MAPE, MAE, and RMSE. The results are shown in Table 2.

The maximum and minimum values of errors in the table are highlighted in bold. It can be seen that the minimum errors appear in the first step, and the maximum errors appear in the 10th step. It shows that the prediction errors increase with the increases of the delay step, which is consistent with the initial expectation. It can be found that, with the increase of the backtracking step, the average increase of the three errors of each delay step is 8.54%, 7.61%, and 8.19%, respectively. In other words, the cost of each additional delay step is that the prediction errors increase by about 8.11%, which is basically within the acceptable range. The multistep prediction results of the LSTM model are shown in Figure 7. Judging from the trend of the indicator curve in the figures, the trends of three error metrics are completely consistent and show an approximate linear growth relationship. This result is in line with common sense, because of the increase of delay steps, there will be a break between the time series and the predicted target, and the interdependence will weaken accordingly. Therefore, it can be concluded that as the number of delay steps increases, the response error of the model shows an approximate linear growth relationship within a limited number of steps.

Another detector with different time sampling data was used for robustness analysis. After evaluating the prediction results, we can see approximate results. The MAE curve of the comparative experiment is shown in Figure 8.

In order to verify the influence of the number of delay steps on the prediction results of other models, the same
processes were performed to predict and evaluate several frequently used models with the 5-minute data of the detector 16,885. The number of backtracking step $L_b$ was set to 12. The number of delay step $D_s$ was gradually increased from 1 to 10. The MAE of the predicted results is shown in Table 3. The numbers in bold are the MAE of the best performing model in each column.

From the data in the table, it can be seen that the minimum values of the MAE of each prediction model appear in the first step, and the number of delay steps is the least at this time. The maximum values all appear in the 10th step, which shows that the prediction errors increase as the number of delay steps increases. It can be seen that as the number of delay steps increases, the average increase in MAE for each delay step is about 9% to 10%.

Similarly, the MAPEs of multistep prediction are plotted in Figure 9. It can be seen that as the prediction delay steps increase, there is a certain difference in MAPE value between all models, but they all show almost the same linear growth trend, which is precisely the same as the previous conclusions. We can also roughly see the accuracy of different traffic flow models in forecasting. The k-nearest neighbour (KNN) model is located at the top of the graph, far away from other models, and has the worst prediction accuracy among all models. In the middle are other machine learning and deep learning models including SVM, RF, GBDT, DT.

| Errors | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   | 10  |
|--------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| MAPE   | 4.91| 6.09| 6.87| 7.51| 8.12| 8.58| 8.99| 9.36| 9.89| 10.12|
| MAE    | 2.13| 2.57| 2.84| 3.08| 3.32| 3.48| 3.59| 3.79| 3.97| 4.08 |
| RMSE   | 3.38| 4.32| 4.82| 5.26| 5.63| 5.89| 6.09| 6.35| 6.57| 6.73 |

Figure 7: Multistep prediction errors of LSTM model on detector 16,885 with 5-minute interval. (a) MAPE. (b) MAE. (c) RMSE.

Figure 8: Multistep prediction MAEs of LSTM model on detector 16,395 with different time intervals. (a) 5-minute interval. (b) 10-minute interval. (c) 15-minute interval.
The models with the best prediction performance are at the bottom of figure. They are a multilayer neural network model and three RNN models including SAE, LSTM, BiLSTM, and GRU. Although the differences among these models are very small, in most cases, BiLSTM’s performances are still better than those of the first three. Among them, when the number of delay steps is small, BiLSTM and LSTM models are comparable, but, after the last 8 steps, BiLSTM defeats LSTM by virtue of the number of neurons being twice that of LSTM. This also fully shows that the BiLSTM model has great potential in the task of time series prediction of traffic conditions. Nevertheless, these three variants of RNNs stand out from many models, which also shows that the recurrent neural networks have incomparable advantages in mining causality of time series and are suitable for traffic time series prediction and other fields.

5. Results Analysis

In the experiment, actual traffic flow speed data was selected to verify the prediction performance of the time series model. Primarily, the number of backtracking steps of the LSTM model was searched through the layer-by-layer greedy coordinate descent method. With different detector data, the influences of backtracking steps on the prediction accuracy of time series were verified, discovering the law of model accuracy decline. Considering the universality of the model and the complexity of calculations, the reasonable range of backtracking steps is determined, which provides a unified data basis for the measurement of prediction accuracy and the horizontal comparison between different models.

The experiment also tested and analysed the number of delay steps for the time series prediction and found an approximately linear relationship between the prediction accuracy and the number of delay steps within a limited time step. Despite the increase of the number of delay steps, the time gap between the current time point and the predicted time point causes the interruption of the dependency relationship and also makes the error metrics increase to varying degrees. But this kind of delay prediction still has a certain meaning in some applications and can be regarded as an effective alternative. For example, if we want to know the traffic flow after 30 minutes, we can directly train a prediction model with a 30-minute interval for next-step delay prediction, or a 15-minute model for two-step delay prediction, and even use a 5-minute model for six-step delay prediction, and so on. These differences between the multistep delay model and the conventional single step prediction. One is that the accuracy of the prediction results is different, and the prediction accuracy of the model with the smaller delay steps is better. The second is that the scale of the prediction results is different. The scale of the multistep prediction result depends on the original scale of the model, and the sampling scale of the output result is the same as the sampling scale of the input data. Although there will be some loss in prediction accuracy, multistep delay prediction can still improve the diversity of prediction results and provide a more reliable application.

6. Conclusion

After discussing the importance of data sequence in traffic state prediction, this paper creates a time backtracking and multistep delay model based on RNNs to learn and extract long-term and short-term dependencies. A case study was conducted to search and determine the optimal backtracking length of the traffic sequence. Three variants of
RNNs and six frequently used models were subjected to multistep delay prediction to verify the impact of the delay step on the prediction accuracy. The main findings are as follows:

The prediction accuracy decreases monotonously with the increase of the number of backtracking steps and then slowly rises after reaching a relatively low point. Although it is not possible to prove whether this relatively low point is the global minimum, the number of backtracking steps corresponding to the first low point can bring the best model performance and maintain a more reasonable computational consumption.

There is an approximately linear relationship between the prediction accuracy of the model and the delay step in the finite time step. Obviously, as the number of delay steps increases, there are multiple time intervals between the current time point and the predicted time point, and the dependence in the sequence will also be interrupted, resulting in a corresponding increase in the error metrics of the model. However, this multistep delay prediction method can still play a special role in some applications, and it also increases the diversity of prediction results.

Comparing the prediction performance of six frequently used models, it is found that three RNN models stand out in all models, which shows that recurrent neural networks have unparalleled advantages in time series causality mining and are very suitable for time series data prediction.

In the future, we will continue our research from three aspects: firstly, traffic flow patterns exhibit strong randomness and uncertainty due to external influences. There are many factors that affect the accuracy of predictions, such as weather, environment, and social activities. Therefore, more factors need to be considered to improve the accuracy of prediction. Secondly, more intelligent algorithms (e.g., evolutionary computing or genetic algorithms) will be tried to optimize the number of backtracking steps. Finally, the ideas of time backtracking and multistep delay can be extended to other fields and provide inspiration and new research directions.

Data Availability

The traffic state data used to support the findings of this study were supplied by DRIVE Net (http://www.uwdrive.net). The processed data are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Natural Science Basic Research Program of Shaanxi (Program no. 2020M-258) and the National Natural Science Foundation of China (61672026).

References

[1] M. Zhou, Y. Yu, and X. Qu, “Development of an efficient driving strategy for connected and automated vehicles at signalized intersections: a reinforcement learning approach,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 1, pp. 433–443, 2019.
[2] R. C. Shit, “Crowd intelligence for sustainable futuristic intelligent transportation system: a review,” *IET Intelligent Transport Systems*, vol. 14, no. 6, pp. 480–494, 2020.
[3] X. Xu, Y. Liu, W. Wang et al., “TTS-frame: a framework for multi-aspect analysis in the field of intelligent transportation systems,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 8, pp. 2893–2902, 2019.
[4] P. Delle Site, “A mixed-behaviour equilibrium model under predictive and static Advanced Traveler Information Systems (ATIS) and state-dependent route choice,” *Transportation Research Part C: Emerging Technologies*, vol. 86, pp. 549–562, 2018.
[5] X. Qu, Y. Yu, M. Zhou, C.-T. Lin, and X. Wang, “Jointly dampening traffic oscillations and improving energy consumption with electric, connected and automated vehicles: a reinforcement learning based approach,” *Applied Energy*, vol. 257, Article ID 114030, 2020.
[6] D. Ma, X. Song, and P. Li, “Daily traffic flow forecasting through a contextual convolutional recurrent neural network modelling inter-and intra-day traffic patterns,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 99, pp. 1–10, 2020.
[7] L. Qu, W. Li, W. Li, D. Ma, and Y. Wang, “Daily long-term traffic flow forecasting based on a deep neural network,” *Expert Systems with Applications*, vol. 121, pp. 304–312, 2019.
[8] W. Zhang, Y. Feng, K. Lu, Y. Song, and Y. Wang, “Speed prediction based on a traffic factor state network model,” *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2020.
[9] X. Song, W. Li, D. Ma, D. Wang, L. Qu, and Y. Wang, “A match-then-predict method for daily traffic flow forecasting based on group method of data handling,” *Computer-Aided Civil and Infrastructure Engineering*, vol. 33, no. 11, pp. 982–998, 2018.
[10] P. Khandelwal, J. Konar, and B. Brahma, “Training RNN and its variants using sliding window technique,” in *Proceedings of the 2020 IEEE International Students’ Conference on Electrical, Electronics and Computer Science (SCEECS)*, pp. 1–5, IEEE, Bhopal, India, 2020.
[11] D. Yang, K. Chen, M. Yang, and X. Zhao, “Urban rail transit passenger flow forecast based on LSTM with enhanced long-term features,” *IET Intelligent Transport Systems*, vol. 13, no. 10, pp. 1475–1482, 2019.
[12] Z. Cui, R. Ke, Z. Pu, and Y. Wang, “Stacked bidirectional and unidirectional LSTM recurrent neural network for forecasting network-wide traffic state with missing values,” 2020, http://arxiv.org/abs/2005.11627.
[13] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, “Long short-term memory neural network for traffic speed prediction using remote microwave sensor data,” *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 187–197, 2015.
[14] J. Cheng, G. Li, and X. Chen, “Research on travel time prediction model of freeway based on gradient boosting decision tree,” *IEEE Access*, vol. 7, pp. 7466–7480, 2018.
[15] J. Tang, X. Chen, Z. Hu, F. Zong, C. Han, and L. Li, “Traffic flow prediction based on combination of support vector
machine and data denoising schemes,” *Physica A: Statistical Mechanics and its Applications*, vol. 534, Article ID 120642, 2019.

[16] H. Yu, N. Ji, Y. Ren, and C. Yang, “A special event-based K-nearest neighbor model for short-term traffic state prediction,” *IEEE Access*, vol. 7, pp. 8177–81729, 2019.

[17] R. P. Magalhaes, F. Lettich, J. A. Macedo et al., “Speed prediction in large and dynamic traffic sensor networks,” *Information Systems*, Article ID 101444, 2019.

[18] X. Zhan, S. Zhang, W. Y. Szeto, and X. Chen, “Multi-step-ahead traffic speed forecasting using multi-output gradient boosting regression tree,” *Journal of Intelligent Transportation Systems*, vol. 24, no. 2, pp. 125–141, 2020.

[19] H. Jiang, Y. Zou, S. Zhang, J. Tang, and Y. Wang, “Short-term speed prediction using remote microwave sensor data: machine learning versus statistical model,” *Mathematical Problems in Engineering*, vol. 2016, Article ID 9326156, 2016.

[20] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[21] W. Huang, G. Song, H. Hong, and K. Xie, “Deep architecture for traffic flow prediction: deep belief networks with multitask learning,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 2191–2201, 2014.

[22] T. Zhou, G. Han, X. Xu et al., “6-agree AdaBoost stacked autoencoder for short-term traffic flow forecasting,” *Neurocomputing*, vol. 247, pp. 31–38, 2017.

[23] W. Zhang, Y. Yu, Y. Qi, F. Shu, and Y. Wang, “Short-term traffic flow prediction based on spatio-temporal analysis and CNN deep learning.” *Transportmetrica A: Transport Science*, vol. 15, no. 2, pp. 1688–1711, 2019.

[24] J. Tang, F. Liu, Y. Zou, W. Zhang, and Y. Wang, “An improved fuzzy neural network for traffic speed prediction considering periodic characteristic,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 9, pp. 2340–2350, 2017.

[25] D. Ma, B. Sheng, S. Jin, X. Ma, and P. Gao, “Short-term traffic flow forecasting by selecting appropriate predictions based on pattern matching,” *IEEE Access*, vol. 6, pp. 75629–75638, 2018.

[26] Y. Yu, X. Si, C. Hu, and J. Zhang, “A review of recurrent neural networks: LSTM cells and network architectures,” *Neural Computation*, vol. 31, no. 7, pp. 1235–1270, 2019.

[27] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang, “Deep learning with long short-term memory for time series prediction,” *IEEE Communications Magazine*, vol. 57, no. 6, pp. 114–119, 2019.

[28] Y. Tian, K. Zhang, J. Li, X. Lin, and B. Yang, “LSTM-based traffic flow prediction with missing data,” *Neurocomputing*, vol. 318, pp. 297–305, 2018.

[29] Z. Zhao, W. Chen, X. Wu, P. C. Y. Chen, and J. Liu, “LSTM network: a deep learning approach for short-term traffic forecast,” *IET Intelligent Transport Systems*, vol. 11, no. 2, pp. 68–75, 2017.

[30] D. Zhang and M. R. Kabuka, “Combining weather condition data to predict traffic flow: a GRU-based deep learning approach,” *IET Intelligent Transport Systems*, vol. 12, no. 7, pp. 578–585, 2018.

[31] G. Dai, C. Ma, and X. Xu, “Short-term traffic flow prediction method for urban road sections based on space-time analysis and GRU,” *IEEE Access*, vol. 7, pp. 143025–143035, 2019.

[32] D. Ma, B. Sheng, X. Ma, and S. Jin, “Fuzzy hybrid framework with dynamic weights for short-term traffic flow prediction by mining spatio-temporal correlations,” *IET Intelligent Transport Systems*, vol. 14, no. 2, pp. 73–81, 2019.

[33] L. Zhang, G. Zhu, P. Shen et al., “Learning spatiotemporal features using 3DCNN and convolutional LSTM for gesture recognition,” in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 3120–3128, Venice, Italy, October 2017.

[34] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[35] M. Sajjad and S. Kwon, “Clustering-based speech emotion recognition by incorporating learned features and deep BiLSTM,” *IEEE Access*, vol. 8, pp. 79861–79875, 2020.

[36] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, “A survey of deep neural network architectures and their applications,” *Neurocomputing*, vol. 234, pp. 11–26, 2017.

[37] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[38] F. A. Gers, J. Schmidhuber, and Fred Cummins, *Learning to forget: continual prediction with LSTM*, in *Proceedings of the 9th International Conference on Artificial Neural Networks: ICANN’99*, pp. 850–855, Edinburgh, UK, 1999.

[39] K. Greff, R. K. Srivastava, J. Koutník et al., “LSTM: a search space odyssey,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2016.

[40] R.-H. Zhang, Z.-C. He, H.-W. Wang, F. You, and K.-N. Li, “Study on self-turning tyre friction control for developing main-servo loop integrated chassis control system,” *IEEE Access*, vol. 5, pp. 6649–6660, 2017.

[41] C. Xu, Y. Yang, S. Jin, Z. Qu, and L. Hou, “Potential risk and its influencing factors for separated bicycle paths,” *Accident Analysis & Prevention*, vol. 87, pp. 59–67, 2016.

[42] W. Jiang and S. Siddiqui, “Hyper-parameter optimization for support vector machines using stochastic gradient descent and dual coordinate descent,” *EURO Journal on Computational Optimization*, vol. 8, no. 1, pp. 85–101, 2020.

[43] T. T. Wu and K. Lange, “Coordinate descent algorithms for lasso penalized regression,” *The Annals of Applied Statistics*, vol. 2, no. 1, pp. 224–244, 2008.

[44] E. Liossis, “Efficient serial and parallel SVM training using coordinate descent,” in *Proceedings of the 2013 IEEE Symposium on Computational Intelligence for Engineering Solutions (CIES)*, pp. 76–83, IEEE, Singapore, 2013.

[45] A. M. Abdolhosseini-Qomi, N. Yazdani, and M. Asadpour, “Overlapping communities and the prediction of missing links in multiplex networks,” *Physica A: Statistical Mechanics and its Applications*, vol. 554, Article ID 124650, 2020.