Journal of Urban Development and Management
https://www.acadlore.com/journals/JUDM

An Integrated Convolutional Neural Network-Bidirectional Long Short-Term Memory-Attention Mechanism Model for Enhanced Highway Traffic Flow Prediction

Haoyuan Kan[1*], Chang Li[1], Ziqi Wang[2]

[1] School of Management and Engineering, Capital University of Economics and Business, 100070 Beijing, China
[2] Faculty of Engineering and Information Technology, The University of Melbourne, 1446535 Melbourne, Australia

[*] Correspondence: Haoyuan Kan (22023210030@cueb.edu.cn)

Received: 01-23-2024 **Revised:** 02-28-2024 **Accepted:** 03-04-2024

Citation: H. Y. Kan, C. Li and Z. Q. Wang, "An integrated Convolutional Neural Network-Bidirectional Long Short-Term Memory-Attention Mechanism model for enhanced highway traffic flow prediction," *J. Urban Dev. Manag.*, vol. 3, no. 1, pp. 18–33, 2024. https://doi.org/10.56578/judm030102.

Abstract: The burgeoning expansion of the Internet of Things (IoT) technology has propelled Intelligent Traffic Systems (ITS) to the forefront of IoT applications, with accurate highway traffic flow prediction models playing a pivotal role in their development. Such models are essential for mitigating highway traffic congestion, reducing accident rates, and informing city planning and traffic management strategies. Given the inherent periodicity, non-linearity, and variability of highway traffic data, an innovative model leveraging a Convolutional Neural Network (CNN), Bidirectional Long Short-Term Memory (BiLSTM), and Attention Mechanism (AM) is proposed. In this model, feature extraction is accomplished via the CNN, which subsequently feeds into the BiLSTM for processing temporal dependencies. The integration of an AM enhances the model by weighting and fusing the BiLSTM outputs, thereby refining the prediction accuracy. Through a series of experiments and the application of diverse evaluation metrics, it is demonstrated that the proposed CNN-BiLSTM-AM model surpasses existing models in prediction accuracy and explainability. This advancement positions the model as a significant contribution to the field, offering a robust and insightful tool for highway traffic flow prediction.

Keywords: Traffic flow prediction; Convolutional Neural Network (CNN); Bidirectional Long Short-Term Memory (BiLSTM); Attention Mechanism (AM)

1 Introduction

In today's society, the field of transportation is facing many severe challenges. With the development of the economy and logistics, as well as people's higher requirements for material life, the number of motor vehicles in various countries is increasing every year (see Figure 1 for an example of China), and the incidence of road congestion is increasing year by year. Compared with ordinary urban traffic roads, expressways have the characteristics of being relatively closed, fast speed, and large traffic flow, which assume the main role of traffic between cities. The operation efficiency of highways affects the country's economic level and people's quality of life [1]. Once congestion occurs, it will seriously affect the traffic capacity. It is more practical and effective to build a reasonable and effective traffic flow prediction model instead of expanding existing roads and rebuilding the roads based on the predicted future traffic flow.

Traffic flow refers to the number of participants actually participating in traffic through a certain place or section of the road in a unit time, which is a basic indicator to measure the traffic state of the road. Dynamic management of the traffic network depends on proper short- and mid-term forecasting of traffic states [2]. Predicting short-term traffic flow helps estimate travel time and the existing level of service on a route [3]. In the past, traffic managers were limited in their ability to formulate and deploy reactive traffic response plans to deal with traffic congestion [4]. If the traffic flow of the traffic section in the future can be predicted in advance, the traffic management department can better guide the traffic vehicles and improve the traffic efficiency of the highway. And the model can be used to select the time of less traffic flow to maintain the highway and to rebuild and expand the future highway that does not match the traffic flow in advance. A reasonable and accurate highway traffic flow prediction model is not only an important basis for the daily management and work plan of the traffic management department but also a key part of the national transportation network.

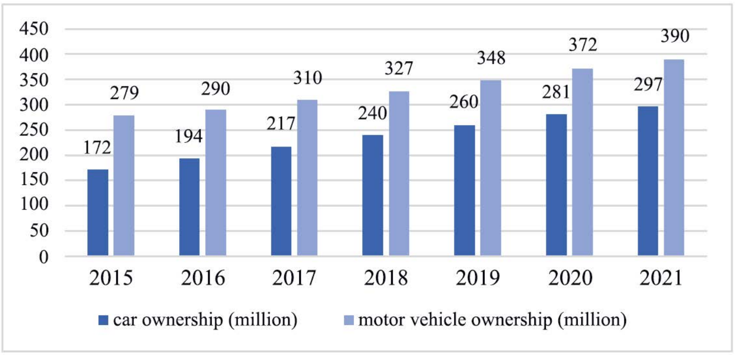


Figure 1. Statistics of motor vehicle ownership in China

Nowadays, many scientists predict highway traffic flow through various algorithms, using the search mechanism of the K-nearest neighbor method to reconstruct the historical traffic flow time series that is similar to the current traffic state and then using the principle of support vector regression to achieve short-term traffic flow prediction [5]. The methods of using artificial neural network models to predict short-term traffic flow [6] all have the problem of low prediction accuracy. The urban highway flow prediction model that integrates graph convolutional neural networks and generative adaptive neural networks (GCN-GAN) has weak interpretability [7]. The traffic flow prediction model that integrates LSTM and GCN can only achieve short-term prediction [8]. The prediction method of average vehicle speed on urban roads based on CNN-LSTM cannot fully consider the impact of other factors such as weather and holidays on traffic flow [9], making it difficult to apply in real traffic environments [10]. At present, considerable research has been conducted on traffic flow prediction models by previous researchers, and each model has its own unique features, providing this paper with many ideas.

Although researchers have explored many methods for predicting traffic flow, existing models still have many problems in predicting highway traffic flow in large samples, high-dimensional data, and complex environments, making it difficult to achieve the expected prediction results. This paper aims to propose a new highway traffic flow prediction model that improves the accuracy and precision of prediction.

By analyzing the models and methods used by predecessors in predicting traffic flow on highways, it is not difficult to see that existing methods mainly use linear models and shallow machine learning models to predict the incoming traffic flow and cannot describe the non-linearity and uncertainty well due to the stochastic and non-linear nature of traffic flow [11]. Compared with the single model, the combined model makes good use of the merits of single models, and its prediction accuracy is higher and fitting ability is stronger. Therefore, this paper aims to build a combined model through CNN, BiLSTM, and AM, select the optimal combination by analyzing the model training results of various combinations of single models and the advantages and disadvantages of various evaluation indicators, and use this as the main model to study the prediction of traffic flow. The remainder of this paper is organized as follows: Section 2 introduces the experimental principles and methods as well as the model; Section 3 analyzes the experimental results; and Section 4 presents the conclusions and limitations of this study, as well as future research directions.

2 Materials and Methods

2.1 Theoretical Basis

2.1.1 CNN

The basic structure of a CNN roughly includes: input layer, convolutional layer, pooling layer, fully connected layer, output layer, as shown in Figure 2.

The structure of CNN is relatively flexible, and the convolutional and pooling layers that belong to feature extraction can be stacked and used. The common combination is several convolutional and pooling layers. The structure of CNN is relatively flexible, and the convolutional and pooling layers that belong to feature extraction can be stacked and used. The common combination is several convolutional and pooling layers. The LeNet framework for handwritten digit recognition, developed based on the CNN model, consists of three convolutional layers and two pooling layers.

The most important concept in convolutional layers is the convolution kernel, which can be understood as a type of feature. The result of multiplying the input and convolution kernel is the projection of the input onto that feature, which can be called a feature map. Taking image recognition as an example, assuming there is a feature representing the contour of an object, multiplying the input image with this feature yields the contour map of the image. The convolution process is shown in subgraph (a) of Figure 3.

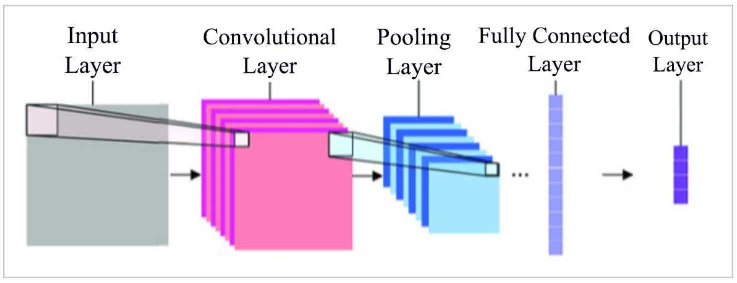


Figure 2. Basic structure of CNN

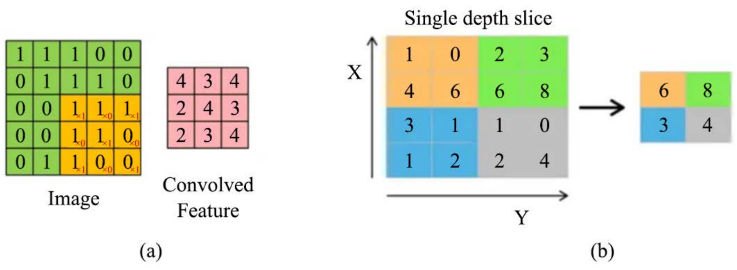


Figure 3. CNN convolutional and pooling layers

Express the convolution process using mathematical formulas as follows:

$$S(i,j) = (x * w)(i,j) + b = \sum_{k=1}^{n\_\text{in}} (x_k * w_k)(i,j) + b \tag{1}$$

Among them, n_in is the number of input matrices or the dimension of the last dimension of the tensor. x_k represents the k-th input matrix. The k-th subconvolutional kernel matrix representing the convolution kernel. *S(i,j)* is the value of the corresponding position element in the output matrix corresponding to the convolutional kernel w.

For the convolutional output, the ReLU activation function is usually used to convert the element values corresponding to positions less than 0 in the output tensor to 0.

The role of the pooling layer is to down-sample the output of the convolutional layer, which mainly includes Max pooling and mean pooling. It is possible to further reduce the data while maintaining the original salient features, as shown in subgraph (b) of Figure 3.

The features obtained through convolution and pooling layers are classified in the fully connected layer. The fully connected layer is a fully connected neural network. Each neuron feedbacks in different proportions based on its weight. Finally, the classification result is obtained by adjusting the weight and network.

Previous studies have shown that one-dimensional CNN has strong feature extraction capabilities for time-series data [12]. Considering the temporal nature of traffic flow data, this paper adopts a one-dimensional CNN model for feature extraction.

2.1.2 LSTM

LSTM is a special type of recurrent neural network (RNN), commonly used for dynamically capturing time dependencies in data [13]. During training, the original RNN is prone to gradient explosion or vanishing as the training time increases and the number of network layers increases, resulting in the inability to process longer sequence data and obtain information from long-distance data. LSTM solves the above problems [14].

The biggest difference between LSTM and RNN lies in their neuron structure, which is distributed in the hidden layer. In LSTM's model, cell states are added to preserve long-term states. The key to LSTM lies in the use of three gating units, namely the forget gate, input gate, and output gate, to update the historical information that needs to be saved and discard irrelevant historical information. The specific process is shown in Figure 4.

The first step of LSTM is to determine what information can be obtained through the cell state. This decision is controlled by the forget gate through sigmoid, which will pass or partially pass based on the output from the previous moment. Its expression is as follows:

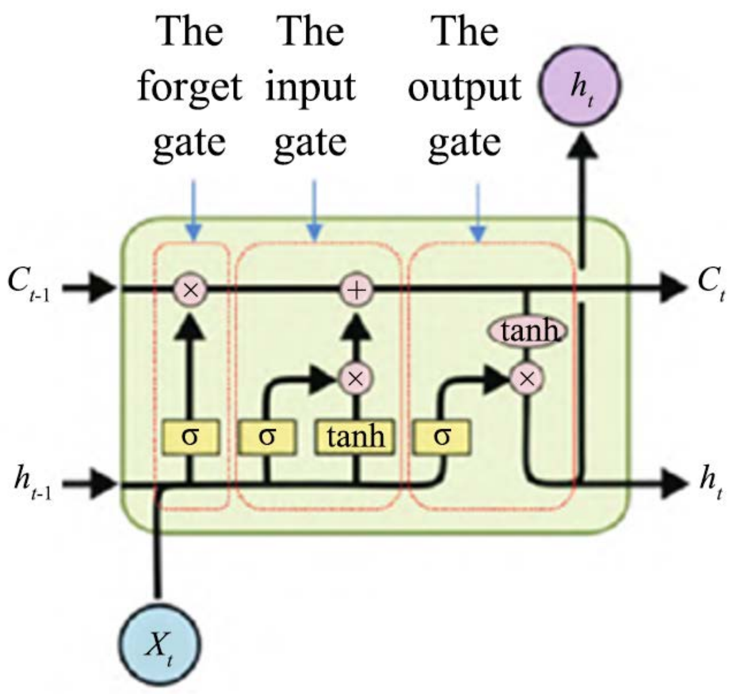


Figure 4. LSTM structure

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{2}$$

In the formula, w_f and b_f are the weight matrix and bias of, h_{t-1} represents the hidden state of the previous moment; x_t represents the input information at the current time.

The second step of LSTM consists of two parts. The first part is to determine which values are used for updating through sigmoid in the input gate. The second part is the tanh layer used to generate new candidate values and add them together to obtain the candidate values. The combination of these two steps is the process of discarding unnecessary information and adding new information, which is expressed as follows:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{3}$$

$$\widetilde{C_t} = \tanh\left(W_c \cdot [h_{t-1}, x_t] + b_c\right) \tag{4}$$

In the formula, f_t, i_t represent the weights of C_{t-1} and $\tilde{C}_t$, C_{t-1} and C_t represent the unit states of the previous and current time, respectively.

The final step of LSTM is to determine the output of the model. Firstly, an initial output is obtained through the sigmoid layer. Then, the tanh function is used to scale the value between -1 and 1, and the output obtained from the sigmoid is multiplied pairwise to obtain the output of the model. Its expression is as follows:

$$O_t = \sigma\left(W_0 \cdot [h_{t-1}, x_t] + b_0\right) \tag{5}$$

$$h_t = o_t \cdot \tanh\left(c_t\right) \tag{6}$$

In the formula, W_0 represents the weight framework of the output layer; b_0 is offset.

The application areas of LSTM include text generation, machine translation, speech recognition, image description generation, and video tagging, but its main application area is the analysis and prediction of time series data.

2.1.3 BiLSTM

The LSTM model uses forward sequence information as input data for time prediction [15], making it difficult to perceive backward data during model training. BiLSTM solves the problem of lacking attention to backward information [16].

Each unit of BiLSTM contains two independent LSTM units, called forward LSTM units and backward LSTM units. The structure of each LSTM unit is the same as in Figure 4, and the two LSTM units are independent of each other. The structure of BiLSTM is shown in Figure 5.

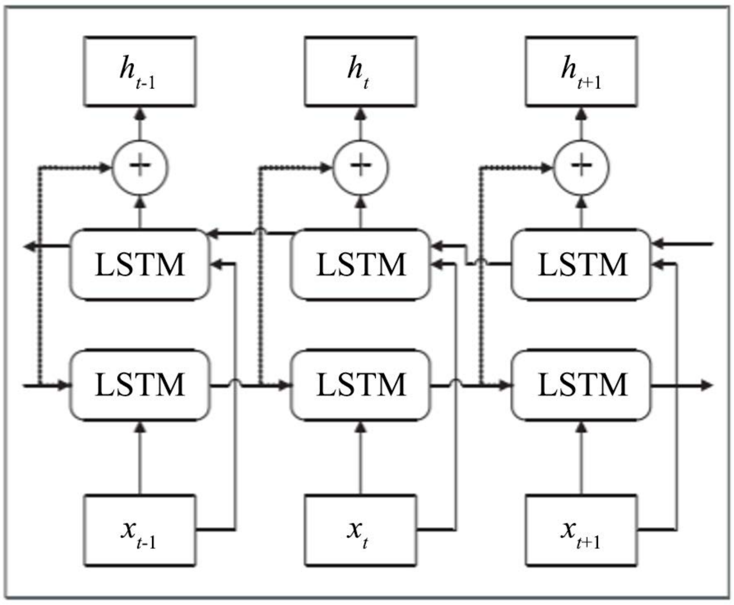


Figure 5. BiLSTM structure

BiLSTM is mainly used in semantic analysis, sentiment analysis, etc. Previous studies have shown that BiLSTM outperforms LSTM in predicting time series data [17].

2.1.4 AM

The AM in neural networks is a resource allocation scheme that allocates computing resources to more important tasks and solves the problem of information overload when computing power is limited [18]. In the process of neural network learning, generally speaking, the more parameters a model has, the stronger its expressive power and the greater the amount of information stored by the model. However, this can lead to the problem of information overload. So, by introducing an AM, focusing on the more critical information for the current task among the numerous input information, reducing attention to other information, and even filtering out irrelevant information, the problem of information overload can be solved, and the efficiency and accuracy of task processing can be improved.

Using $\mathrm{X} = [\mathrm{x}_1, \mathrm{x}_2 \ldots \mathrm{x}_\mathrm{n}]$ to represent n input information, in order to save computational resources, the neural network does not need to process these n inputs information but only needs to select some task-related information from X for calculation. The soft AM refers to the process of selecting information—not just selecting one out of n pieces of information, but calculating the weighted average of n input pieces of information, which are then input into a neural network for calculation. And hard attention refers to selecting information at a certain position in the input sequence, such as randomly selecting a piece of information or selecting the information with the highest probability [19]. Soft attention is generally used to handle neural network problems.

This paper selects the key value pair attention mode from the soft AM. In the key value pair attention mode, the AM is seen as a soft addressing operation that treats input information X as stored content in memory, where each element is composed of an address key and a value. The goal of this operation is to obtain the value corresponding to the query Key=Query from memory, which is the attention value. Unlike traditional hard addressing, in soft addressing, it is not necessary to satisfy the condition of Key=Query in order to retrieve storage information. Instead, the degree of extraction in the element value is determined by calculating the similarity between the query key and the address of the element in the storage, which is called attention weight. The values corresponding to each address will be extracted and then summed, which is equivalent to using the similarity between Query and Key to calculate the weight of each value, and then weighting and summing the values. The final value is the attention value, which is calculated by weighting the similarity between Query and Memory Key.

The above calculation can be divided into three processes (see Figure 6 for the process diagram):

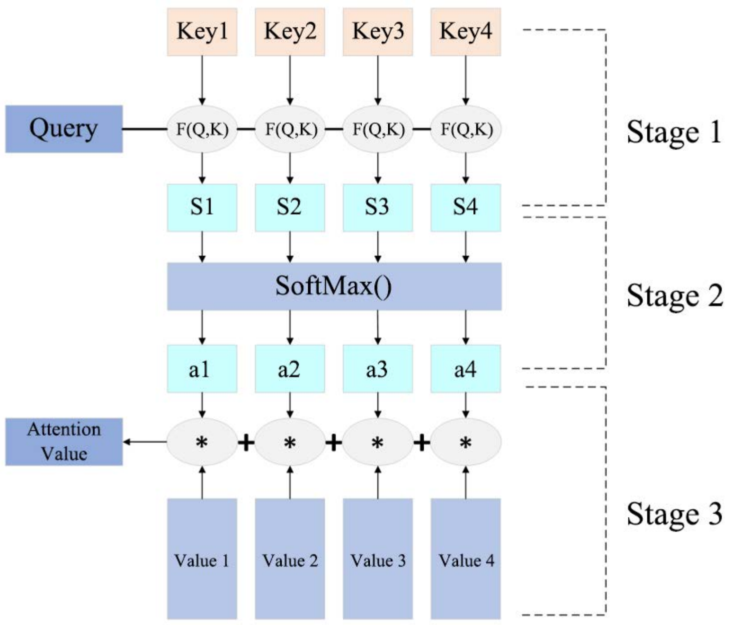


Figure 6. Key value pair attention pattern algorithm structure

1) Calculate the similarity between Query and Key. Attention score X can be calculated using similarity measurement methods such as the additive model, dot product, or mul-ti-layer perceptron (in this model, dot product are used):

$$S_i = F\left(Q, k_i\right) \tag{7}$$

2) Use the SoftMax function to numerically convert attention scores. On the one hand, normalization can be performed to obtain a probability distribution where the sum of all weight coefficients is 1. On the other hand, the characteristics of the SoftMax function can be used to highlight the weights of important elements;

$$a_i = \operatorname{softmax}\left(S_i\right) = \frac{\exp\left(S_i\right)}{\sum_{j=1}^{n} \exp\left(S_i\right)} \tag{8}$$

3) Weighted sum of values based on weight coefficients.

$$\operatorname{Attention}((K, V), Q) = \sum_{i=1}^{n} \alpha_i v_i \tag{9}$$

By combining and organizing the above formulas, we have obtained the key value pair attention pattern algorithm formula:

$$\operatorname{Attention}((K, V), Q) = \sum_{i=1}^{n} \alpha_i v_i = \sum_{i=1}^{n} \frac{\exp\left(s\left(k_i, q\right)\right)}{\sum_{j=1}^{n} \exp\left(s\left(k_i, q\right)\right)} v_i \tag{10}$$

Since the introduction of the AM, it has influenced the development of many artificial intelligence fields based on deep learning algorithms. The current AM has been successfully applied in areas such as image processing, natural language processing, and data prediction, and is in-creasingly closely integrated with other neural networks.

2.2 Experimental Procedure

2.2.1 Data sources and preprocessing

This paper selects the traffic flow data of a detector on the M25 highway in the UK as the dataset. As the busiest outer ring highway in the UK, the data of this highway has great reference value, and the detector is located near Heathrow Airport, with high daily traffic flow and high data reliability. The dataset is divided into three groups, namely the full-year data for 2022, 2021, and 2020. Taking the 2022 dataset as an example, the data was selected from January 1, 2022, to December 31, 2022, with a total of 34848 pieces of data recorded every 15 minutes. The first 80% of the data was used as the model training set, the middle 10% was used as the validation set to determine the network structure and adjust hyperparameters, and the last 10% was used as the test set to evaluate the performance of the final model. The same applies to other datasets.

Next, the time series data will be preprocessed using max min normalization after removing irrelevant columns. The formula is as follows:

$$x' = \frac{x_i - x_{min}}{x_{max} - x_{min}} \tag{11}$$

where, x' represents the normalized data, x_i represents the original data, and x_{max} and x_{min} represent the maximum and minimum values in the dataset, respectively. For outliers in the dataset, simple sum averaging is used to repair the difference.

For time series, taking the 2022 dataset as an example, the date and time columns are integrated and converted to timestamp format. Then, the first-time data, January 1, 2022, at 00:14:00, is used as the starting time as the 0 value, and the other times are normalized to relative time before input.

2.2.2 Model architecture

The model architecture used in this paper is shown in Figure 7.

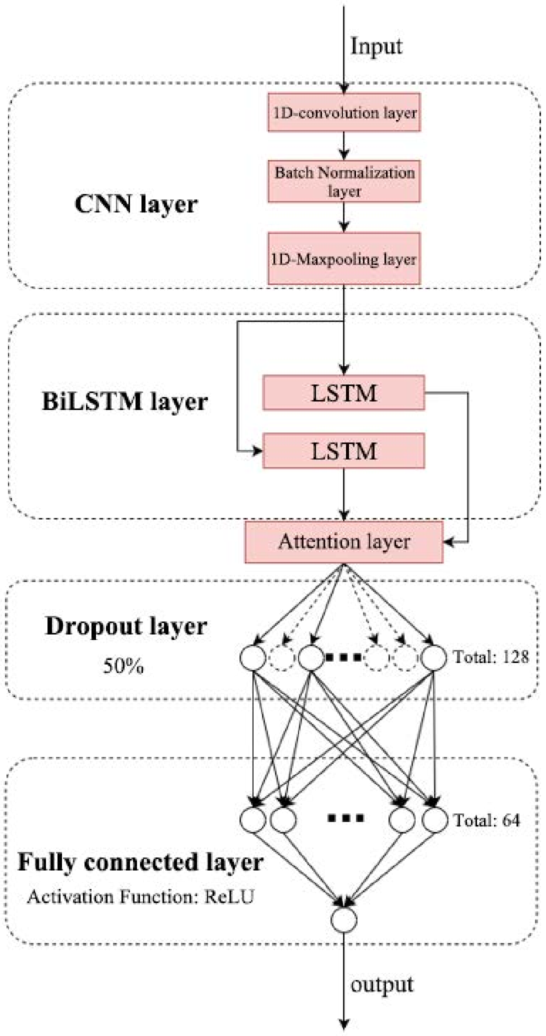


Figure 7. CNN-BiLSTM-AM model architecture

This model consists of five parts: CNN layer, BiLSTM layer, AM layer, Dropout layer, and Fully connected layer.

The CNN layer consists of a one-dimensional convolutional layer, a batch normalization layer, and a one-dimensional max pooling layer. The CNN convolutional layer extracts features from the input data, preprocesses the feature data through the batch normalization layer, and then uses the max pooling layer to reduce the space size of the feature map, reducing the number of parameters and computational costs of the model and outputting it.

The BiLSTM layer is composed of two LSTM units, which are independent of each other. They perform forward and backward processing on the temporal feature data output by the pre-vious layer and output it.

The AM layer performs feature-weighted reoperation on the output data of the previous layer, dividing the features into important features and ordinary features, further enhancing the overall model's perception ability of key information before outputting.

The function of the Dropout layer is to randomly block 50% of neurons without affecting the size of the output dimension, in order to reduce the risk of overfitting and enhance the model's adaptability.

The Fully connected layer consists of two fully connected layers. The first fully connected layer has an activation function of ReLU and an output dimension of 64. The second fully con-nected layer has no activation function and an output dimension of 1.

2.2.3 Hyperparameter optimization

Based on the model training results, optimize the relevant parameters of each layer through grid optimization methods, such as the number of filters in CNN and the number of BiLSTM neurons, to achieve the optimal parameter configuration.

2.2.4 Model training

This paper uses two indicators, mean absolute error (MAE) and mean square error (MSE), as evaluation indicators. The formula is as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|q_i - \hat{q}_i| \tag{12}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(q_i - \hat{q}_i)^2 \tag{13}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(q_i - \hat{q}_i)^2} \tag{14}$$

where, q_i represents the normalized measured traffic flow value, and $\hat{q}_i$ represents the normalized predicted traffic flow value. The lower the MAE, MSE, and RMSE values, the better the predictive performance of the model, and the closer the predicted values are to the true values. The higher the MAE, MSE, and RMSE values, the worse the predictive performance of the model, and the more the predicted values deviate from the true values.

The optimization method of this model adopts the mini-batch gradient descent method. The Adam algorithm is selected as the model, and the loss function MSE, which has good performance in convergence speed and is also an evaluation index, is selected as the loss function. The batch size is set to 32, and the iteration is 100 generations. The learning rate has a significant impact on the convergence process, with an initial setting of 10^{-4}. Two callback functions are defined. The first function is used to reduce the learning rate to one-fifth of its original value when the loss value no longer decreases or the decrease is not significant for five consecutive generations. The second function is used to stop iteration when the loss value no longer decreases or the decrease is not significant for ten consecutive generations to prevent overfitting of the model. The CNN convolution kernel size is set to 1, there are a total of 128 BiLSTM units, the activation function is sigmoid, and the Dropout layer ratio is set to 50%. The AM layer adopts simple weighting processing.

Short-term traffic flow forecasting models rely on the regularity existing in historical data to predict traffic patterns in future time periods [20]. Considering the strong periodicity of the time series data (see Figure 8), it fluctuates on a daily basis. Two parameters, the sine function and the cosine function, were added during data input to reflect periodic characteristics. The formulas are as follows:

$$X_i = \sin\left(2 * \pi * \frac{T_i}{4 * 24}\right) \tag{15}$$

$$Y_i = \cos\left(2 * \pi * \frac{T_i}{4 * 24}\right) \tag{16}$$

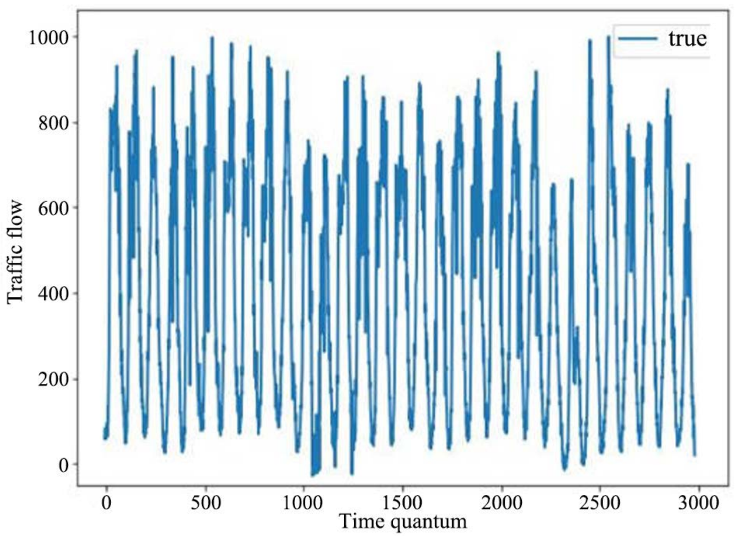


Figure 8. Partial time series data

where, X_i and Y_i represent the sine and cosine parameters of the i-th data, respectively, and T_i represents that the corresponding timestamp of the i-th data is the T-th 15min of the day.

Through this operation, the format of each input data is changed to a ternary output of [relative time, sine parameter, cosine parameter]. For input data with different dates and the same time, the values of the sine and cosine parameters are exactly the same. By adding these two parameters, the ability of CNN to extract periodic features can be increased, making the final prediction results of the model more realistic.

For time series data, smoothing was performed again to eliminate special values.

During the model training process, the various parameters of the model are continuously adjusted based on the performance of the model after multiple iterations, in order to achieve the best performance. When the actual iteration number is around 40, the model already has high accuracy, and the loss value decreases gradually. Figure 9 shows the prediction results of the model on the test set when iterating to the 40th generation.

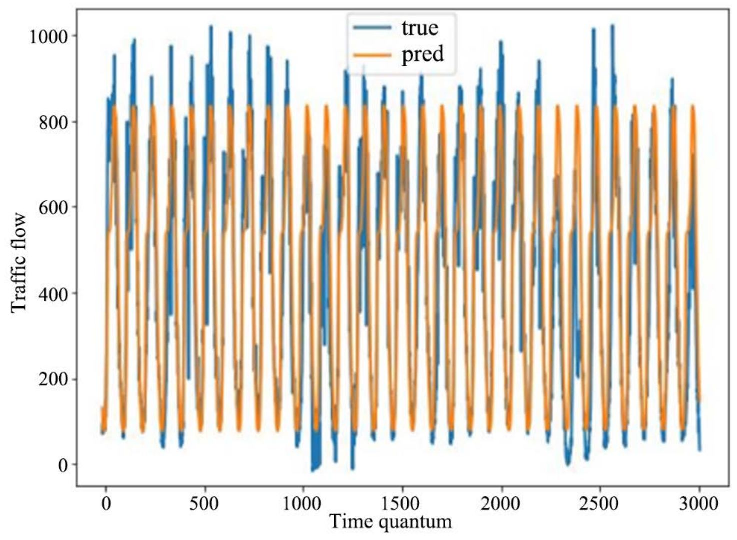


Figure 9. Model prediction results (Dec. 2022 dataset)

Because the test set date is December, there is a significant difference in actual traffic flow compared to other months due to comprehensive factors such as holidays and weather. However, the prediction results of this model still fit the actual values well, except for a few special values, which have basically matched, and there is no underfitting or overfitting phenomenon. This in-dicates that the model has been trained and can be used for evaluation parameter analysis.

3 Result

3.1 Analysis of Evaluation Indicators

Figure 10 shows the convergence degree of the loss function of the CNN-BiLSTM-AM model trained on different datasets, from left to right in 2022, 2021, and 2020, respectively.

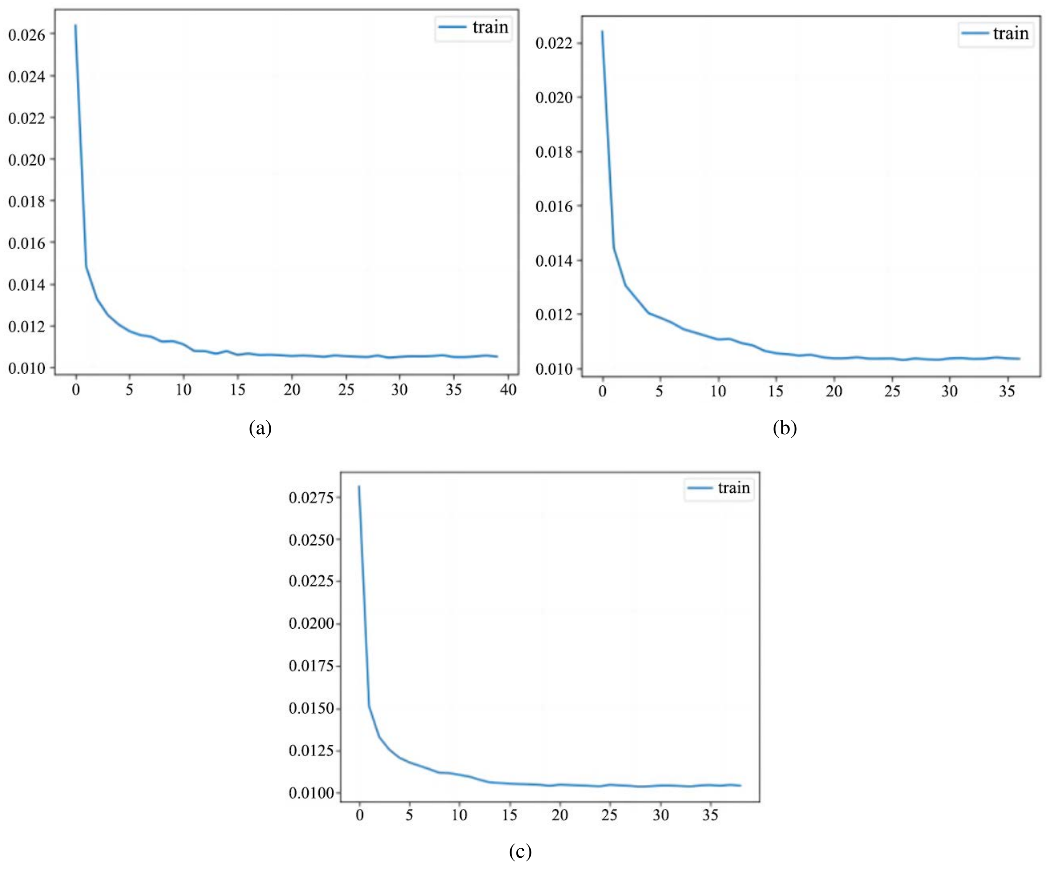


Figure 10. CNN-BiLSTM-AM model loss function

It can be seen that for different datasets, the model can fit well, and the loss function con-tinuously decreases with iteration, tends to stabilize after 25 generations, and gradually reduces the learning rate, further refining the model. Finally, the loss function no longer decreases between 35 and 40 generations, and the model training is completed.

Figure 11 shows the changes in loss function (loss value) and accuracy (acc) of different models during the training process on the 2022 dataset, where DoLSTM and DoBiLSTM represent double-layer LSTM and BiLSTM, respectively.

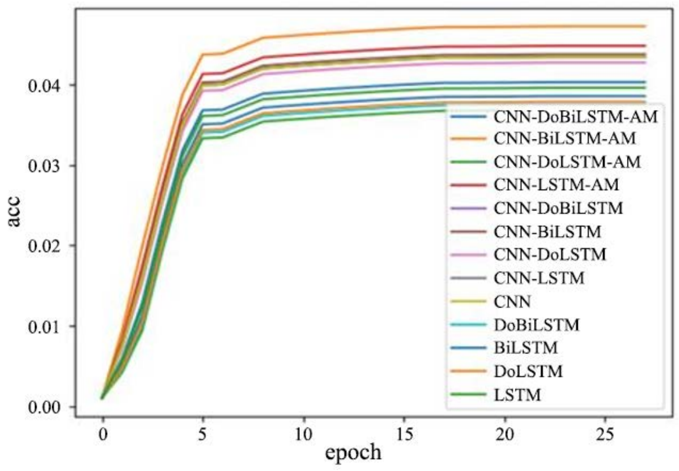
From subgraph (a) of Figure 11, it can be seen that the accuracy of each model increases rapidly with the iteration process and gradually flattens out, with the CNN BiLSTM-AM model having the highest accuracy.

From subgraph (b) of Figure 11, it can be seen that, except for the CNN model with a relatively high initial loss value, the initial loss values of other models are generally consistent. However, as the iteration process progresses, the loss values of each model rapidly decrease.

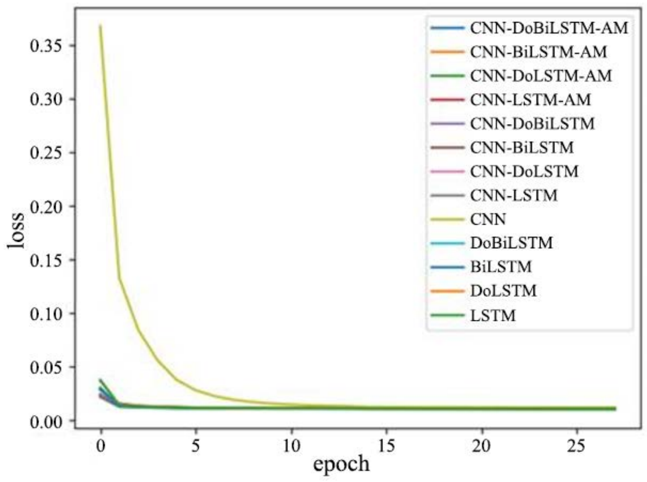
From subgraph (c) of Figure 11, it can be seen that models with double-layer BiLSTM or dou-ble-layer LSTM have lower loss values, which indirectly indicates the overfitting defect of these models.

Due to the limitations of the dataset itself, it is difficult for acc to accurately express the accuracy of the model. In reality, traffic flow prediction models can only predict data that is infinitely close to the true value through training optimization. Therefore, in order to better demonstrate the accuracy of each model, this article defines the accuracy of the model as: if the predicted traffic flow from the test set does not fluctuate more than 80-120% of the actual traffic flow, it is considered that the model has successfully predicted, and based on this, the accuracy of each model can be obtained (as shown in Table 1).

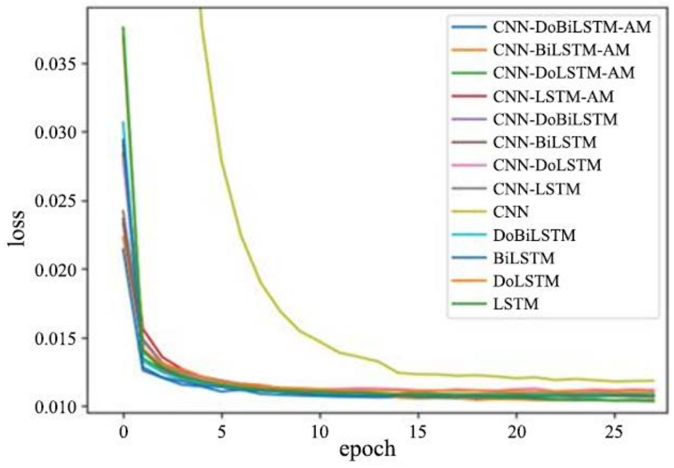
According to Table 1, the CNN-BiLSTM-AM model has the highest accuracy, and for models with an added AM mechanism, the double-layer LSTM significantly reduces the accuracy of the model, and the accuracy of a single model is also lower than that of a combination model.



(a)



(b)



(c)

Figure 11. Performance of each model on the 2022 dataset: (a) Changes in ACC values during each model iteration process; (b) Changes in loss values during each model iteration process; (c) Changes in loss values during iteration of models other than CNN

Table 2, Table 3 and Table 4 show the evaluation parameters of the CNN-BiLSTM-AM model and other com-binations or single models trained on the 2022, 2021, and 2020 datasets, respectively.

MAE, MSE, and RMSE in the table represent the evaluation indicators of the training set; Val_MAE, Val_MSE, and Val_RMSE represent the evaluation metrics of the validation set. Other models were used as control groups, and the architecture was modified based on the model presented in this paper. Taking the CNN bilayer BiLSTM-AM model as an example, an additional BiLSTM layer and Dropout layer were added between the BiLSTM and AM layers on the basis of the model presented in this paper. The number of neurons in the additional BiLSTM layer was 128, and the Dropout layer ratio was set to 50%. The hyperparameters of the other layers were also consistent with the model presented in this paper.

Table 1. Accuracy of various models (from high to low)

Model	Accuracy
CNN-BiLSTM-AM	**72.31%**
CNN-LSTM-AM	71.98%
CNN-LSTM	70.75%
BiLSTM	69.69%
CNN-DoLSTM	69.22%
DoBiLSTM	67.67%
DoLSTM	67.50%
LSTM	67.10%
CNN-DoBiLSTM	66.03%
CNN-BiLSTM	64.82%
CNN	60.16%
CNN-DoBiLSTM-AM	43.81%
CNN-DoLSTM-AM	31.75%

Table 2. Various model evaluation parameters (2022 dataset)

Model	MAE	MSE	RMSE	Val_MAE	Val_MSE	Val_RMSE
CNN-BiLSTM-AM	**0.0741**	**0.0104**	**0.1020**	**0.0860**	**0.0182**	**0.1349**
CNN-DoBiLSTM-AM	0.0723	0.0100	0.1000	0.1455	0.0453	0.2128
CNN-DoLSTM-AM	0.0738	0.0102	0.1010	0.1858	0.0609	0.2468
CNN-LSTM-AM	0.0748	0.0105	0.1025	0.0872	0.0192	0.1386
CNN-DoBiLSTM	0.0752	0.0106	0.1030	0.0872	0.0195	0.1396
CNN-BiLSTM	0.0756	0.0107	0.1034	0.0883	0.0193	0.1389
CNN-DoLSTM	0.0748	0.0105	0.1025	0.0866	0.0192	0.1386
CNN-LSTM	0.0753	0.0106	0.1030	0.0872	0.0187	0.1367
CNN	0.0765	0.0111	0.1054	0.0868	0.0188	0.1371
DoBiLSTM	0.0761	0.0109	0.1044	0.0940	0.0233	0.1526
BiLSTM	0.0742	0.0104	0.1020	0.0886	0.0204	0.1428
DoLSTM	0.0762	0.0109	0.1044	0.0943	0.0233	0.1526
LSTM	0.0754	0.0107	0.1034	0.0946	0.0229	0.1513

From the table, it can be seen that for the three datasets, the CNN-BiLSTM-AM model per-forms relatively stable with small changes in evaluation parameters, indicating strong applicability of the model.

Taking the 2022 dataset as an example, Table 2 shows that adding double-layer BiLSTM or double-layer LSTM to the composite model can increase the fitting degree of the data to a certain extent. For example, the MSE of the CNN double-layer BiLSTM-AM model is slightly lower than that of the CNN BiLSTM-AM model, but at the same time, for the Val of the CNN double-layer BiLSTM-AM model, the MSE is much higher than that of the CNN-BiLSTM-AM model, indi-cating that the double-layer BiLSTM or double-layer LSTM structure leads to overfitting of the model and is not suitable for actual traffic flow prediction models.

Replacing the LSTM layer with the BiLSTM layer and adding an AM layer to the model both reduce the evaluation metrics of the training and validation sets, indicating that BiLSTM out-performs LSTM in processing time series data and combining with CNN, AM, and other models. Additionally, the addition of the AM layer enhances the predictive performance of the model.

Compared with the combined model, the single-model CNN, LSTM, and BiLSTM all perform poorly. CNN only focuses on the extraction of data features, ignoring the time series of data, which makes the evaluation indicators of its training set and verification set high. Although LSTM and BiLSTM value the temporal nature of data, they lack the ability to extract features. This results in both performing well on the training set and experiencing severe overfitting and mediocre performance on the validation set, making it difficult to achieve predictive results. In contrast, BiLSTM has a slightly lower overfitting degree than LSTM, and single-layer LSTM and BiLSTM have a slightly lower overfitting degree than double-layer LSTM and BiLSTM.

Table 3. Various model evaluation parameters (2021 dataset)

Model	MAE	MSE	RMSE	Val_MAE	Val_MSE	Val_RMSE
CNN-BiLSTM-AM	**0.0762**	**0.0109**	**0.1044**	**0.0860**	**0.0180**	**0.1342**
CNN-DoBiLSTM-AM	0.0736	0.0102	0.1010	0.1682	0.0546	0.2337
CNN-DoLSTM-AM	0.0742	0.0103	0.1015	0.1925	0.0645	0.2540
CNN-LSTM-AM	0.0752	0.0106	0.1030	0.0876	0.0183	0.1353
CNN-DoBiLSTM	0.0763	0.0109	0.1044	0.0877	0.0193	0.1389
CNN-BiLSTM	0.0757	0.0107	0.1034	0.0875	0.0190	0.1378
CNN-DoLSTM	0.0759	0.0107	0.1034	0.0874	0.0194	0.1393
CNN-LSTM	0.0765	0.0109	0.1044	0.0881	0.0182	0.1349
CNN	0.0815	0.0121	0.1100	0.0879	0.0192	0.1386
DoBiLSTM	0.0757	0.0108	0.1039	0.0940	0.0232	0.1523
BiLSTM	0.0741	0.0104	0.1020	0.0892	0.0207	0.1439
DoLSTM	0.0772	0.0112	0.1058	0.0945	0.0234	0.1530
LSTM	0.0756	0.0107	0.1034	0.0939	0.0226	0.1503

Table 4. Various model evaluation parameters (2020 dataset)

Model	MAE	MSE	RMSE	Val_MAE	Val_MSE	Val_RMSE
CNN-BiLSTM-AM	**0.0746**	**0.0105**	**0.1025**	**0.0854**	**0.0181**	**0.1345**
CNN-DoBiLSTM-AM	0.0732	0.0102	0.1010	0.1523	0.0474	0.2177
CNN-DoLSTM-AM	0.0736	0.0103	0.1015	0.1672	0.0528	0.2298
CNN-LSTM-AM	0.0751	0.0105	0.1025	0.0910	0.0185	0.1360
CNN-DoBiLSTM	0.0752	0.0107	0.1034	0.0866	0.0189	0.1375
CNN-BiLSTM	0.0750	0.0105	0.1025	0.0877	0.0192	0.1386
CNN-DoLSTM	0.0776	0.0111	0.1054	0.0882	0.0193	0.1389
CNN-LSTM	0.0757	0.0107	0.1034	0.0882	0.0187	0.1367
CNN	0.0769	0.0112	0.1058	0.0876	0.0194	0.1393
DoBiLSTM	0.0750	0.0107	0.1034	0.0945	0.0232	0.1523
BiLSTM	0.0747	0.0106	0.1030	0.0926	0.0222	0.1490
DoLSTM	0.0762	0.0109	0.1044	0.0944	0.0234	0.1530
LSTM	0.0758	0.0108	0.1039	0.0932	0.0225	0.1500

Both the training and validation sets show that the proposed model can achieve lower MAE and MSE values, indicating good training performance, high prediction accuracy, and strong applicability. Moreover, compared with a single model, the CNN-BiLSTM-AM model can take into account both the bidirectional time series and feature extraction, and can also focus on more important features in the time series, which is impossible for any single model.

3.2 Analysis of Evaluation Indicators

Robustness, where robustness is a transliteration of Robust, meaning robust and robust. It is also the ability of the system to survive in abnormal and dangerous situations. In a neural network model, robustness can be understood as the model's tolerance to data changes. Huber provides three requirements for robustness from the perspective of robust statistics.

1) The model has high accuracy or effectiveness.

2) For small deviations in model assumptions, it only has a minor impact on the performance of the algorithm.

3) For large deviations in model assumptions that do not have a catastrophic impact on al-gorithm performance.

For 1), this article has been demonstrated through evaluation indicators in 3.1, so this section will focus on analyzing 2) and 3) to evaluate the robustness of the model (using the 2022 dataset as an example).

3.2.1 Small deviation experiment

To measure the impact of small deviations on the model, this article adds simple noise to all time series data in the 2022 dataset, that is, adding a random integer with an interval of -5 to 5 to each data point, and ensuring that each data point is still greater than 0 after addition. Take the data with the added noise as input, train the model, and analyze the changes in the evaluation indicators of each model compared to the original evaluation indicators. Take two decimal places as a percentage, with positive values representing the increase compared to the original value and negative values representing the decrease compared to the original value.

Table 5. Changes in evaluation indicators of various models with small deviations (2022 dataset)

Model	MAE	MSE	RMSE	Val_MAE	Val_MSE	Val_RMSE
CNN-BiLSTM-AM	**-0.54%**	**-0.96%**	**-9.80%**	**0.23%**	**4.40%**	**20.98%**
CNN-DoBiLSTM-AM	2.07%	3.00%	17.32%	15.12%	18.10%	42.54%
CNN-DoLSTM-AM	0.54%	0.98%	9.90%	0.65%	1.97%	14.04%
CNN-LSTM-AM	0.80%	0.95%	9.75%	-1.38%	-0.54%	-7.35%
CNN-DoBiLSTM	1.20%	1.89%	13.75%	-0.57%	-2.56%	-16.00%
CNN-BiLSTM	-1.59%	-2.80%	-16.73%	-1.59%	-0.52%	-7.21%
CNN-DoLSTM	4.68%	6.67%	25.83%	3.00%	1.56%	12.49%
CNN-LSTM	0.13%	0.00%	0.00%	0.23%	-1.60%	-12.65%
CNN	3.92%	6.31%	25.12%	5.18%	9.04%	30.07%
DoBiLSTM	-0.13%	0.00%	0.00%	-0.43%	-0.43%	-6.56%
BiLSTM	1.21%	1.92%	13.86%	3.16%	7.35%	27.11%
DoLSTM	0.39%	0.92%	9.59%	-0.74%	-0.43%	-6.56%
LSTM	0.53%	0.93%	9.64%	-0.74%	-1.31%	-11.45%

As shown in Table 5, the CNN-BiLSTM-AM model has a small variation amplitude when facing small deviations in input data, and can still ensure the accuracy of the model.

3.2.2 Large deviation experiment

To measure the impact of large deviations on the model, this article adds simple noise to all time series data in the 2022 dataset, which means adding a random integer with an interval of -100 to 100 to each data point. It is not guaranteed that each data point will still be greater than 0 after addition. Take the data with the added noise as input, train the model, and analyze the changes in the evaluation indicators of each model compared to the original evaluation indicators. Take two decimal places as a percentage, with positive values representing the increase compared to the original value and negative values representing the decrease compared to the original value.

Table 6. Changes in evaluation indicators of various models with large deviations (2022 dataset)

Model	MAE	MSE	RMSE	Val_MAE	Val_MSE	Val_RMSE
CNN-BiLSTM-AM	**11.74%**	**13.46%**	**36.69%**	**9.30%**	**0.00%**	**0.00%**
CNN-DoBiLSTM-AM	14.11%	17.00%	41.23%	19.86%	23.18%	48.15%
CNN-DoLSTM-AM	11.25%	13.73%	37.05%	-7.75%	-13.63%	-36.92%
CNN-LSTM-AM	13.37%	17.14%	41.40%	11.93%	-1.61%	-12.69%
CNN-DoBiLSTM	12.90%	16.04%	40.05%	8.94%	-2.56%	-16.00%
CNN-BiLSTM	10.58%	12.15%	34.86%	7.25%	-1.04%	-10.20%
CNN-DoLSTM	14.84%	20.00%	44.72%	10.05%	0.52%	7.21%
CNN-LSTM	11.95%	15.09%	38.85%	8.14%	-4.28%	-20.69%
CNN	12.81%	14.41%	37.96%	8.64%	-1.60%	-12.65%
DoBiLSTM	11.30%	12.84%	35.83%	6.91%	-3.86%	-19.65%
BiLSTM	11.86%	13.46%	36.69%	9.14%	-0.49%	-7.00%
DoLSTM	12.47%	15.60%	39.50%	5.94%	-5.15%	-22.69%
LSTM	11.27%	12.15%	34.86%	6.45%	-3.93%	-19.82%

According to Table 6, it can be seen that the CNN-BiLSTM-AM model has a relatively small change in amplitude when facing large deviations in input data, and can still ensure the normal operation of the model.

In summary, the CNN-BiLSTM-AM model has high accuracy and can ensure its accuracy is not greatly affected when there are small errors in the input data. It can still operate normally when there are large deviations in the input data. Therefore, its robustness is good compared to other models, making it a mature and robust model.

4 Conclusion

In view of the characteristics of traffic flow series, this paper proposes a combined model of CNN, BiLSTM, and AM, which combines the advantages of CNN, BiLSTM, and AM. The ex-ample verification shows that the model performs well in three sets of data sets. Compared with other single models and similar combined models, the evaluation index of the model is more excellent, and the robustness is good. It is a robust and mature model. Replacing the LSTM layer with the BiLSTM layer and adding the AM layer both improve the prediction accuracy and ap-plicability of this model. Compared with other models, this model can better process traffic flow data.

However, this model still has many shortcomings. Due to the limitations of the dataset, this model only considers time as a feature. However, in reality, weather, holidays, other road condi-tions, emergencies, and even the daily flight situation of Heathrow Airport near the detector may affect the traffic flow of that road segment at a single time. These factors could be quantified and analyzed as input features for extraction, followed by weighted reprocessing using AM. For example, quantifying weather factors using one-hot encoding as one of the inputs to the model.

At the same time, the dataset can also be expanded by obtaining data from other detectors on the expressway and the location of the detector map, in order to simulate the traffic flow situation of the entire section of the expressway. By analyzing and predicting the traffic flow situation of each section of the expressway in real time, the applicability of the model can be further enhanced, and the practical application ability of the model can be further enhanced. Increase the spatial dependency of the model. The improved model can become an important basis for traffic man-agement departments to manage highways and also participate in road construction.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] Z. J. Zheng, Z. L. Wang, L. Y. Zhu, and H. Jiang, "Determinants of the congestion caused by a traffic accident in urban road networks," *Accid. Anal. Prev.*, vol. 136, p. 105327, 2020. https://doi.org/10.1016/j.aap.2019.105327

[2] Y. Zhang and Y. C. Liu, "Traffic forecasting using least squares support vector machines," *Transportmetrica*, vol. 5, no. 3, pp. 193–213, 2009. https://doi.org/10.1080/18128600902823216

[3] K. A. Momin, S. Barua, M. S. Jamil, and O. F. Hamim, "Short duration traffic flow prediction using Kalman Filtering," in *AIP Conference Proceedings, Khulna, Bangladesh*, vol. 2713, no. 1. AIP Publishing, 2023. https://doi.org/10.1063/5.0129721

[4] W. W. Yue, C. L. Li, G. Q. Mao, N. Cheng, and D. Zhou, "Evolution of road traffic congestion control: A survey from perspective of sensing, communication, and computation," *China Commun.*, vol. 18, no. 12, pp. 151–177, 2021. https://doi.org/10.23919/JCC.2021.12.010

[5] W. Q. Zhuang and Y. B. Cao, "Short-term traffic flow prediction based on a K-Nearest Neighbor and bidirectional Long Short-Term Memory model," *Appl. Sci.*, vol. 13, no. 4, p. 2681, 2023. https://doi.org/10.3390/app13042681

[6] S. V. Kumar and L. Vanajakshi, "Short-term traffic flow prediction using seasonal ARIMA model with limited input data," *Eur. Transp. Res. Rev.*, vol. 7, pp. 1–9, 2015. https://doi.org/10.1007/s12544-015-0170-8

[7] H. L. Zheng, X. Li, Y. F. Li, Z. Q. Yan, and T. H. Li, "GCN-GAN: Integrating graph convolutional network and generative adversarial network for traffic flow prediction," *IEEE Access*, vol. 10, pp. 94 051–94 062, 2022. https://doi.org/10.1109/ACCESS.2022.3204036

[8] Z. Z. Wu, M. X. Huang, A. P. Zhao, and Z. X. Lan, "Traffic prediction based on GCN-LSTM model," in *Journal of Physics: Conference Series*, vol. 1972, no. 1. IOP Publishing, 2021, p. 012107.

[9] K. Zhao, S. X. Yuan, Z. Z. Wang, and J. X. Wang, "Research on urban road mean speed prediction method based on LSTM-CNN model," in *2022 IEEE 7th International Conference on Intelligent Transportation Engineering (ICITE), Beijing, China*, 2022, pp. 365–371. https://doi.org/10.1109/ICITE56321.2022.10101481

[10] M. M. Cao, V. O. K. Li, and V. W. S. Chan, "A CNN-LSTM model for traffic speed prediction," in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring), Antwerp, Belgium*, 2020, pp. 1–5. https://doi.org/10.1109/VTC2020-Spring48590.2020.9129440

[11] X. D. Ran, Z. G. Shan, Y. Shi, and C. Lin, "Short-term travel time prediction: A spatiotemporal deep learning approach," *Int. J. Inf. Technol. Decis. Mak.*, vol. 18, no. 04, pp. 1087–1111, 2019. https://doi.org/10.1142/S0219622019500202

[12] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1D convolutional neural networks and applications: A survey," *Mech. Syst. Signal Process.*, vol. 151, p. 107398, 2021. https://doi.org/10.1016/j.ymssp.2020.107398

[13] A. Baghbani, N. Bouguila, and Z. Patterson, "Short-term passenger flow prediction using a bus network graph convolutional long short-term memory neural network model," *Transp. Res. Rec.*, vol. 2677, no. 2, pp. 1331–1340, 2023. https://doi.org/10.1177/03611981221112673

[14] Y. X. Hua, Z. F. Zhao, R. P. Li, X. F. Chen, Z. M. Liu, and H. G. Zhang, "Deep learning with long short-term memory for time series prediction," *IEEE Commun. Mag.*, vol. 57, no. 6, pp. 114–119, 2019. https://doi.org/10.1109/MCOM.2019.1800155

[15] T. Bogaerts, A. D. Masegosa, J. S. Angarita-Zapata, E. Onieva, and P. Hellinckx, "A graph CNN-LSTM neural network for short and long-term traffic forecasting based on trajectory data," *Transp. Res. Part C Emerg. Technol.*, vol. 112, pp. 62–77, 2020. https://doi.org/10.1016/j.trc.2020.01.010

[16] H. Zhang, G. Yang, H. L. Yu, and Z. Zheng, "Kalman Filter-based CNN-BiLSTM-ATT model for traffic flow prediction," *Comput. Mater. Contin.*, vol. 76, no. 1, 2023. https://doi.org/10.32604/cmc.2023.039274

[17] S. Siami-Namini, N. Tavakoli, and A. S. Namin, "The performance of LSTM and BiLSTM in forecasting time series," in *2019 IEEE International Conference on Big Data, Los Angeles, CA, USA*, 2019, pp. 3285–3292. https://doi.org/10.1109/BigData47090.2019.9005997

[18] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014. https://doi.org/10.48550/arXiv.1409.0473

[19] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *Proceedings of the 32nd International Conference on Machine Learning*, 2015, pp. 2048–2057. https://doi.org/10.1049/IET-WSS.2019.0106

[20] J. J. Tang, F. Liu, Y. J. Zou, W. B. Zhang, and Y. H. Wang, "An improved fuzzy neural network for traffic speed prediction considering periodic characteristic," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 9, pp. 2340–2350, 2017. https://doi.org/10.1109/TITS.2016.2643005