Mathematical Biosciences and Engineering

MBE, 20(11): 19617–19635.
DOI: 10.3934/mbe.2023869
Received: 09 August 2023
Revised: 10 October 2023
Accepted: 11 October 2023
Published: 26 October 2023

http://www.aimspress.com/journal/MBE

Research article

Research on transformer and long short-term memory neural network car-following model considering data loss

Pinpin Qin[1,*], Xing Li[1], Shenglin Bin[1], Fumao Wu[1] and Yanzhi Pang[2]

[1] School of Mechanical Engineering, Guangxi University, Nanning 530004, China
[2] Guangxi Key Laboratory of International Join for China-ASEAN Comprehensive Transportation, Nanning University, Nanning 530200, China

* **Correspondence:** Email: qpinpin@gxu.edu.cn.

Abstract: There is limited research on the loss and reconstruction of car-following features. To delve into car-following's characteristics, we propose a car-following model based on LSTM-Transformer. By fully leveraging the advantages of long short-term memory (LSTM) and transformer models, this study focuses on reconstructing the input car-following features. Training and testing were conducted using 700 car-following segments extracted from a natural driving dataset and the Next Generation Simulation (NGSIM) dataset, and the proposed model was compared with an LSTM model and an intelligent driver model. The results demonstrate that the model performs exceptionally well in feature reconstruction. Moreover, compared to the other two models, it effectively captures the car-following features and accurately predicts the position and speed of the following car when features are lost. Additionally, the LSTM-Transformer model accurately reproduces traffic phenomena, such as asymmetric driving behavior, traffic oscillations and lag, by reconstructing the lost features. Therefore, the LSTM-Transformer car-following model proposed in this study exhibits advantages in feature reconstruction and reproducing traffic phenomena compared to other models.

Keywords: car-following; car-following features; long short-term memory; transformer; reconstruction

1. Introduction

The car-following model is an important research direction in traffic flow theory, traffic control, intelligent transportation systems, etc. The research outcomes in this area have significant theoretical

and practical implications for improving traffic flow efficiency, reducing traffic accidents and enhancing the overall traffic environment. Existing car-following models can be classified into theory-driven and data-driven models [1]. The theory-driven model proposes specific model structures based on reasonable assumptions regarding vehicle interactions, aiming to capture the mutual influences between the states of two vehicles. However, it cannot quantify certain human factors, has high complexity and limited accuracy [2,3]. The data-driven car-following model employs data-driven techniques to perform statistical analysis on car-following data, thereby discovering underlying driving behavior patterns and establishing corresponding decision-making or mapping relationships to predict car-following behaviors. The deep-learning car-following model is an extension of the data-driven car-following model. It introduces more complex neural network structures and algorithms on top of the data-driven model to better handle complicated relationships and nonlinear features in car-following data. In addition, the physics-informed car-following models aim to integrate the advantages of these two models. These models combine physical principles and deep learning to improve the understanding and prediction ability of car-following behavior and have high interpretability and generalization performance while dealing with uncertainties [4–6].

The calibration and validation of the car-following model heavily rely on complete and detailed vehicle trajectory data, as these data contain the car-following features that the model needs to learn. However, due to the characteristics of data collection and recording techniques, real-world vehicle trajectory data often suffer from inaccuracies and missing information, which can lead to the loss of the car-following features and severely impact the performance of the car-following model. Previous research has recognized vehicle trajectory data's imperfections and employed various data recovery methods to reconstruct the trajectory data. Montanino and Punzo [7] demonstrated that the accuracy of empirical vehicle trajectory influences driver model behavior. They proposed a "traffic-informed" method to restore trajectory completeness and validated its effectiveness through simulations. The results indicate that trajectory accuracy impacts driver heterogeneity, and reconstructing trajectory parameters can better reproduce macroscopic congestion patterns. Wang et al. [8] proposed a linear interpolation model that considers temporal and spatial correlations and an online calibration module. Through field data evaluation, the model outperformed previous models that only considered spatial correlations and improved interpolation accuracy. The online calibration is applicable for online interpolation in traffic control. Tak et al. [9] proposed an improved k-nearest neighbors (KNN) approach for data imputation of road segments, considering spatial and temporal correlations. This method simultaneously handles missing data from multiple related sensors, improving computational efficiency. The method enhances imputation accuracy by preserving the geometric attributes of each road segment. Even in cases where the missing data type is unidentified or a mixture of missing types, the algorithm still demonstrates accurate and stable imputation performance. Chiou et al. [10] addressed the issue of missing data by sampling daily traffic flow trajectories from stochastic functions and leveraging the data features of functional data analysis. They proposed using conditional expectation methods for imputing missing values in functional principal component analysis (FPCA). Tang et al. [11] proposed a method based on fuzzy c-means (FCM) for data imputation in loop detectors. By considering spatiotemporal correlations, the data structure was transformed into matrix form, and genetic algorithms were utilized to optimize the parameters of the FCM model. The results demonstrated that the FCM-based imputation method outperformed traditional methods. Zhao et al. [12] proposed a tensor completion method based on multimodal characteristics to recover missing data. To address the issue of data sparsity, virtual sensor nodes were introduced, and two-dimensional linear interpolation and piecewise estimation methods were applied to estimate the average travel time between nodes. Furthermore, an optimal k-nearest

neighbors (KNN) method was proposed for travel time prediction. The results indicated that the proposed method improved data quality and prediction accuracy. Duan et al. [13] proposed a denoising stacked autoencoder for traffic data imputation. Efficient deep learning-based imputation was achieved by considering both temporal and spatial factors. The excellent performance of deep learning in traffic data imputation was revealed by visualizing the hidden layer features. The study demonstrated the effectiveness and efficiency of deep learning in traffic data imputation and analysis. Zhuang et al. [14] proposed an innovative traffic data imputation method, where the original data was first transformed into spatiotemporal images. Deep learning methods were then applied to the images using a convolutional neural network (CNN)-based context encoder to reconstruct the complete image from the missing data. The results demonstrated that this novel method improved imputation accuracy and exhibited a stable error distribution. Zhao et al. [15] addressed the issue of missing values in vehicle trajectory data by proposing a novel imputation adversarial convolutional neural network (IACNN) method. The IACNN method utilizes preceding vehicle trajectories to impute consecutive missing data. The results demonstrate that compared to traditional methods, the proposed model performs better in handling consecutive data losses and holds significant application value in traffic modeling and simulation. Liang et al. [16] introduced a novel deep learning framework called memory-augmented dynamic graph convolution network (MDGCN) for imputing missing traffic data. The model utilizes a recurrent layer to capture temporal information and a graph convolution layer to capture spatial information. Extensive experiments were conducted using two publicly available traffic speed datasets. The results demonstrate that the proposed model outperforms existing state-of-the-art deep learning methods in various missing data scenarios. Zhao et al. [17] proposed a deep learning framework to impute missing trajectory data called map-embedded graph attention network (TrajGAT). The results demonstrate that TrajGAT exhibits excellent performance on evaluation metrics. Existing methods overly rely on individual vehicle trajectory data when reconstructing trajectory data, and there is a lack of research on reconstructing trajectories by incorporating both the ego vehicle and the preceding vehicle's following trajectory data.

The main objective of this study is to propose a car-following model capable of reconstructing the car-following features and integrating them with the deep learning-based car-following model. The aim is to provide an efficient and accurate car-following model with practical significance in traffic flow theory, traffic control and intelligent transportation systems. In this paper, we propose a car-following model based on LSTM-Transformer. This model combines the strengths of LSTM and transformer, maintaining good car-following performance even in the case of continuous loss of car-following features. To validate the model, we conducted tests to evaluate its reconstruction effectiveness. Furthermore, we compared and analyzed the performance of the LSTM-Transformer model in terms of prediction accuracy and its ability to reproduce traffic phenomena in the presence and absence of car-following features. The remaining sections of this paper are organized as follows: Section 2 introduces the data and methods, Section 3 presents the experiments and relevant discussions and Section 4 summarizes our findings.

2. Data and methodology

2.1. Data

The natural driving database used in this study was captured by the Ubiquitous Traffic Eyes (UTE) team from Southeast University, using UAVs equipped with 4K high-definition cameras to capture the traffic flow on urban highways [18]. The team employed algorithms to extract

high-resolution vehicle trajectory data. Manual correction and validation were performed on several vehicle position deviations to ensure 100% vehicle detection and tracking in the open trajectory dataset. The data used in this paper are from Datasets 1 and 3 of this database. The Next Generation Simulation (NGSIM) I-80 dataset consists of trajectory data from the eastbound I-80 highway in Emeryville, California. It includes three 15-minute videos, and the data selected for this study corresponds to the period from 5:00 to 5:15 p.m. Following the principles outlined below, 700 car-following segments were collectively extracted from the two datasets.

1) During the data processing, the longitudinal distance between the ego vehicle and the preceding vehicle in each car-following segment was limited to less than 80 m to ensure that the ego vehicle was not in free-flow conditions.

2) Each car-following segment was confined to the same lane, and no lane-changing or overtaking maneuvers were present.

3) When extracting the trajectory data for the two vehicles in each car-following segment, the car-following duration must be at least 15 seconds.

2.2. LSTM network

Recurrent neural network (RNN) is a type of neural network that takes the previous time step's output as input for the current time step, enabling the learning of long-term dependencies. However, when dealing with long-term dependencies, RNN needs to avoid vanishing and exploding gradient problems. LSTM, as an improved variant of RNN, effectively addresses these issues [19]. LSTM has gained widespread application in natural language processing, speech recognition and time series prediction thanks to its ability to accurately capture long-term dependencies and mitigate gradient-related problems.

The recurrent network structure is illustrated in Figure 1, where a distinctive "gate" structure is utilized to control the flow of information, enabling the memory and control tasks of the cell state. This ensures continuous updating of the memory within the LSTM unit.

The forget gate (f_t): By determining whether to retain or discard information in the cell state, the forgetting gate performs the task of discarding information that may lead to erroneous predictions. Equations (1) and (2) show the calculation process of discarded information.

$$\sigma\left(x\right)=\frac{1}{1+e^{-x}}, \tag{1}$$

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

where W_f and b_f represent the weight matrix and bias of the forgetting gate, respectively. σ is the activation function, which constrains the transformed values within the range of (0, 1). $\left[h_{t-1},x_t\right]$ represents the matrix formed by combining the previous cell output h_{t-1} and the current input x_t.

The input gate (i_t) is used to update the cell state by storing important information in the cell state, thereby enhancing the data. The Eqs (3) to (6) illustrate the calculation process of updating the cell state.

$$tanh\left(x\right)=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}, \tag{3}$$

$$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right), \tag{4}$$

$$\widetilde{C}_t = tanh\left(W_c \cdot \left[h_{t-1}, x_t\right] + b_c\right), \tag{5}$$

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t, \tag{6}$$

where W_i and b_i represent the weight matrix and bias of the input gate, respectively. W_c and b_c represent the weight matrix and bias of the updated value, respectively. $\widetilde{C}_t$ is the candidate's value in the state. $tanh$ is the activation function for the candidate cell information, which constrains the transformed values within the range of [-1, 1]. C_t represents the updated cell state.

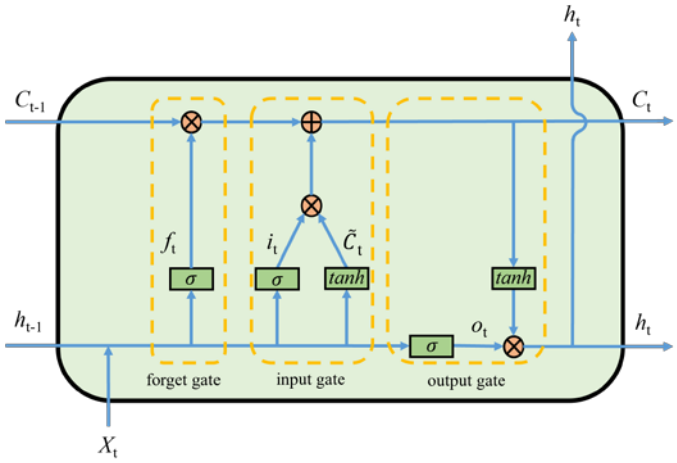


Figure 1. Schematic diagram of the recurrent unit structure in the LSTM network.

The output gate (o_t) determines the information we intend to output based on the cell state. Equations (7) and (8) are the calculation process.

$$o_t = \sigma\left(W_o \cdot \left[h_{t-1}, x_t\right] + b_o\right), \tag{7}$$

$$h_t = o_t * tanh\left(C_t\right), \tag{8}$$

where W_o and b_o represent the weight matrix and bias of the output gate, respectively.

LSTM records the car-following state C_t of the following car at time *t*. The door structure can not only help extract the short-term car-following characteristics of the following car, but also the new cell state can remember the car-following information h_t at each time. The car-following information at each time and the current time is used as the input of the next time to realize the practical mining of the short-term car-following characteristics of the following car.

2.3. Transformer module structure

The network structure of the transformer is mainly composed of attention mechanisms. Its

advantages are strong visibility, simple structure and fast parallel speed, and the transformer is widely used in the natural language processing and image fields [20]. Its structure is shown in Figure 2. The overall architecture consists of four modules: input module, encoding module, decoding module and output module.

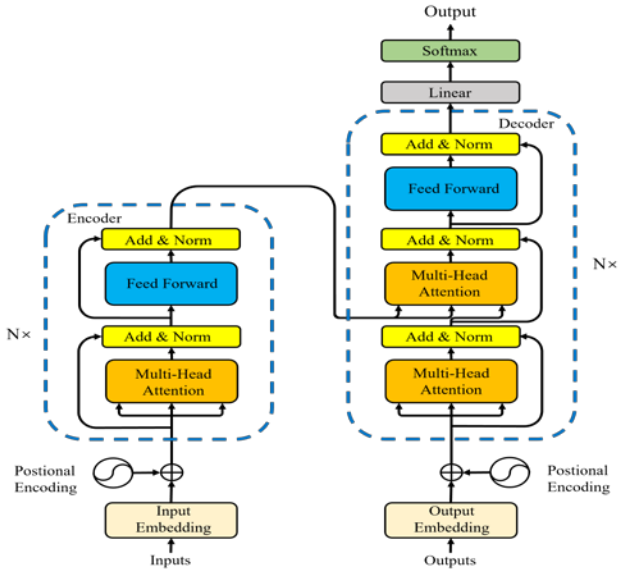


Figure 2. Transformer model structure.

Input module: Because the transformer has no loop structure, it is necessary to encode the position of the input sequence to represent the relative or absolute position relationship of the elements in the sequence. The car-following data feature h_t mined by LSTM is used as input, and the position encoding (PE) encodes the position of h_t. Equations (9)–(11) show the calculation process of coding.

$$PE\left(pos,2i\right)=\sin\left(pos\Big/10000^{\frac{2i}{d_{\mathrm{model}}}}\right), \tag{9}$$

$$PE\left(pos,2i+1\right)=\cos\left(pos\Big/10000^{\frac{2i}{d_{\mathrm{model}}}}\right), \tag{10}$$

$$h_i=Embedding\left(h_t\right)+PE\left(h_t\right), \tag{11}$$

where pos denotes the absolute position in the sequence. $d_{\mathrm{model}}=512$ represents the dimension of data embedding. h_i is the encoding vector.

Encoding module: composed of N encoders, each encoder comprises three parts: multi-head attention mechanism, feedforward neural network, residual connection and normalization layer.

The multi-head attention mechanism assigns the weight to the h_i that completes the position encoding, assigning the high weight to the critical information and the low weight to the secondary

information to realize the deep mining of the nonlinear characteristics of the car-following data. Equations (12)–(15) are the calculation steps of the attention mechanism.

$$Q,K,V = h_i, \tag{12}$$

$$Attention\left(Q,K,V\right) = soft\max\left(QK^T/\sqrt{d_k}\right)V, \tag{13}$$

$$Multihead\left(Q,K,V\right) = concat\left(head_1,\cdots,head_k\right)W^o, \tag{14}$$

$$head_i = Attention\left(QW_i^Q, JW_i^K, VW_i^V\right), \tag{15}$$

where $W_i^Q \in R^{d_{model}\times d_k}$, $W_i^K \in R^{d_{model}\times d_k}$, $W_i^V \in R^{d_{model}\times d_v}$, $W^o \in R^{hd_v\times d_{model}}$ are parameter vectors.

The feedforward neural network reduces the influence of long sequences on the fitting degree of attention mechanism. With the increase in layers, the fitting data may be too small or too large. Adding the planning layer to standardize the data can improve the running speed of the model. The calculation formula is as follows:

$$H_i = LayerNorm\left(h_i + Multihead\left(Q,K,V\right)\right), \tag{16}$$

$$H_i' = FFN\left(H_i\right), \tag{17}$$

$$H_t = LayerNorm\left(H_i + H_i'\right). \tag{18}$$

The decoding module is composed of N decoders and each decoder comprises four parts: multi-head self-attention mechanism, multi-head attention mechanism, feedforward neural network, residual connection and normalization layer.

The output module: Because the input car-following data is numerical data, the linear layer and *softmax* functions can be replaced by *swish* and *sigmoid* functions, which play roles in data normalization.

2.4. LSTM-transformer car-following model

2.4.1. Framework of car-following model

Transformer was initially applied in the field of natural language processing, in which position encoding was introduced to give the relative position information of elements in the sequence of the model. Sun et al. [21] use the transformer-based framework to predict the speed of the front car, and its position encoding can better capture the correlation in the time series data and improve the car-following performance. Therefore, our proposed LSTM-Transformer car-following model takes full advantage of LSTM and transformer to enhance the accuracy of car-following behavior prediction.

The LSTM unit is used as a sequence feature extractor to fully use its advantages in capturing the long-term dependence of time series data to effectively capture the time correlation in the

car-following data sequence. The feature information extracted from LSTM is input to access the subsequent transformer module. To ensure the smooth flow of information between LSTM and transformer components, a linear transformation is performed to adapt its dimensions to the expectations of the transformer architecture. Transformer's encoder is introduced to further process feature information. With its inherent global dependence capture ability, it performs well in analyzing complex sequence data. Each transformer's encoder consists of a multi-head self-attention mechanism and a feedforward neural network, which helps to capture the relationship between features at different locations. To maximize the advantages of LSTM and the transformer, the output of the Transformer's encoder is fused with the output of LSTM, which helps to retain the long-term memory ability of LSTM and the advantages of the transformer in modeling global relationships. The structure is shown in Figure 3. According to the output data of the model, the vehicle state is updated by Eqs (19)–(21) [22].

$$a_n(t) = \left[v_n(t+T) - v_n(t)\right]/T, \tag{19}$$

$$x_n(t+T) = x_n(t) + v_n(t)\cdot T + \frac{1}{2}a_n(t)\cdot T^2, \tag{20}$$

$$\Delta x(t) = x_{n-1}(t) - x_n(t), \tag{21}$$

where a_n is the acceleration of the nth vehicle at time t, T is the time step of sampling (0.08 or 0.1 s), v_n is the speed of the nth vehicle at time t, $x_{n-1}(t)$ and $x_n(t)$ are the longitudinal displacement of the n-1th vehicle and the nth vehicle, respectively.

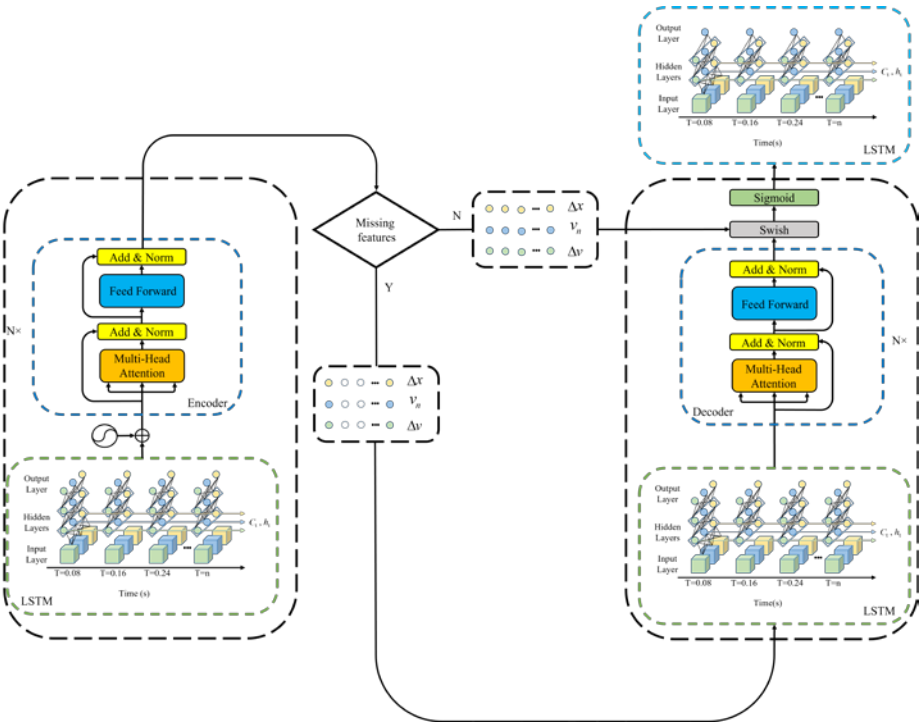


Figure 3. The framework of LSTM-transformer car-following model.

2.4.2. Configuration of the model

Since car-following behavior varies with road types and traffic conditions and the loss of car-following data features, the model is configured according to the observed vehicle trajectory data and road conditions to improve its ability to simulate car-following behavior. The model parameters are set as shown in Table 1.

Table 1. LSTM-Transformer model configuration.

Number of layers	Neuron type	Number of neurons	Activation function
1	LSTM	128	ReLU
2	Transformer	128	ReLU
3	LSTM	128	ReLU
4	Transformer	128	ReLU
5	Dense	12	Swish
6	Dense	1	Sigmoid
7	LSTM	128	ReLU

2.4.3. Input and output variables of the model

During car-following, human driver vehicles (HDVs) adjust their speed by observing the distance between the driving vehicle and the preceding vehicle and the preceding vehicle's speed. For intelligent connected vehicles (CAVs), the displacement and speed relationship between the ego vehicle and the preceding vehicle is also considered to establish a reasonable car-following strategy [23]. To make the model have an excellent ability to reproduce the car-following behavior, the distance between the ego vehicle and the preceding vehicle (Δx), the relative speed (Δv) and the speed of the ego vehicle (v_n) are taken as the input variables and the predicted speed of the ego vehicle (v_{np}) is taken as the output variable.

2.4.4. Metrics of the model performance

To measure the model's prediction accuracy and the reconstruction accuracy of the lost car-following data, we used the mean square error (MSE) as the performance evaluation index of the model.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i^{\text{obs}} - y_i^{\text{sim}}\right)^2, \tag{22}$$

where n is the number of samples and y_i^{obs} and y_i^{sim} are the ith observed and simulated values, respectively.

3. Results and discussion

3.1. Calibration of models

70% of the data is randomly selected to train the LSTM-Transformer car-following model, and the remaining 30% is used for testing. The calibration results of the LSTM-Transformer model parameters are shown in Table 2. The calibration results of the LSTM model parameters are shown in Table 3. The calibration results of the intelligent driver model (IDM) parameters are shown in Table 4.

Table 2. Parameters of the LSTM-Transformer car-following model.

Parameter	Value
N	6
d_{model}	512
d_{k}	64
Memory time step	5
Prediction time step	10
Epochs	40
Batch size	64
Dropout rate	0.1

Table 3. Parameters of the LSTM car-following model.

Parameter	Value
Memory time step	5
Prediction time step	10
Epochs	40
Batch size	32
Dropout rate	0.1

Table 4. Parameters of the IDM.

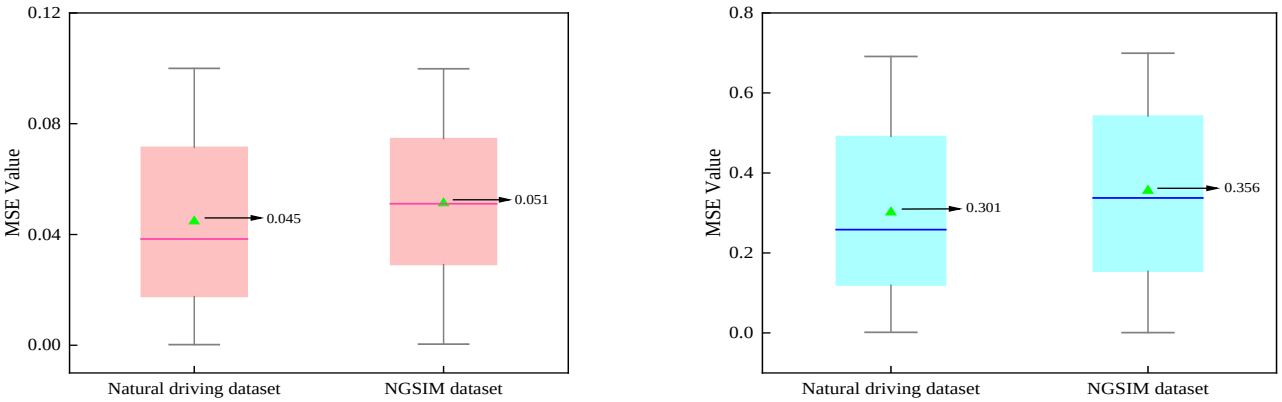
Parameter	Dataset	
	Natural driving dataset	NGSIM dataset
Desired speed (m/s)	[9, 21]	[8, 20]
Maximum acceleration (m/s^2)	[1.1, 2.1]	[1, 2.2]
Comfortable deceleration (m/s^2)	[0.8, 2.3]	[0.9, 2.6]
Time headway (s)	[0.8, 1.8]	[0.6, 1.5]

3.2. Car-following features the reconstruction of the model

3.2.1. Features reconstruction

The car-following model usually regards relative speed and gap as the key features of car-following. For improving road traffic efficiency and safety, reconstructing the lost features is of great practical significance and research value. To ensure the integrity of the car-following features,

the LSTM-Transformer car-following model introduces a feature reconstruction phase to avoid the impact of input feature loss on model performance. Since the cyclic detection of the sensor can achieve a loss rate of about 20% [24], the features that account for 20% of each car-following segment are randomly selected for abandonment and reconstruction. The reconstruction results are evaluated by the performance index MSE. According to the statistical results of different datasets, the reconstructed relative speed is distributed with different mean MSE in other datasets (0.045 for the natural driving dataset and 0.051 for the NGSIM dataset), as shown in Figure 4(a). The reconstruction gap is distributed with different mean MSE in other datasets (0.301 for the natural driving dataset and 0.356 for the NGSIM dataset), as shown in Figure 4(b). The average MSE values of each feature in different datasets are relatively small and similar, indicating that the model has stable performance in reconstructing features. We draw a dataset whose MSE is close to the mean value, as shown in Figure 5. A good consistency between the reconstructed and actual feature points indicates that the model has a good effect on reconstruction.



(a) MSE distribution of reconstruction relative speed (b) MSE distribution of reconstruction gap

Figure 4. The distribution of MSE when the model reconstructs features.

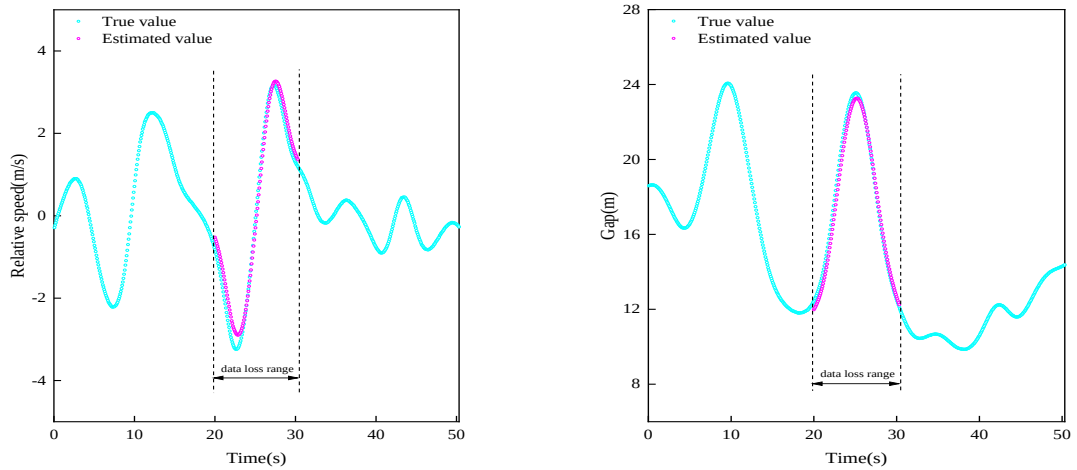


Figure 5. The effect of model reconstruction features.

3.2.2. Comparison of prediction accuracy after reconstruction

To present the performance of the LSTM-Transformer model, we compare the simulated trajectory with the actual trajectory data and the simulated trajectory reproduced by the classical theory-driven model (IDM) and data-driven model (LSTM).

Table 5. Statistical results of speed simulation error.

Dataset	Model	MSE		
		Minimum	Mean	Maximum
Natural driving dataset	IDM	0.092	0.334	1.426
	LSTM	0.048	0.197	0.527
	LSTM-Transformer	0.017	0.079	0.328
	(miss)LSTM-Transformer	0.019	0.092	0.371
NGSIM dataset	IDM	0.189	1.824	8.235
	LSTM	0.156	1.561	6.541
	LSTM-Transformer	0.122	0.745	4.336
	(miss)LSTM-Transformer	0.128	0.783	4.437

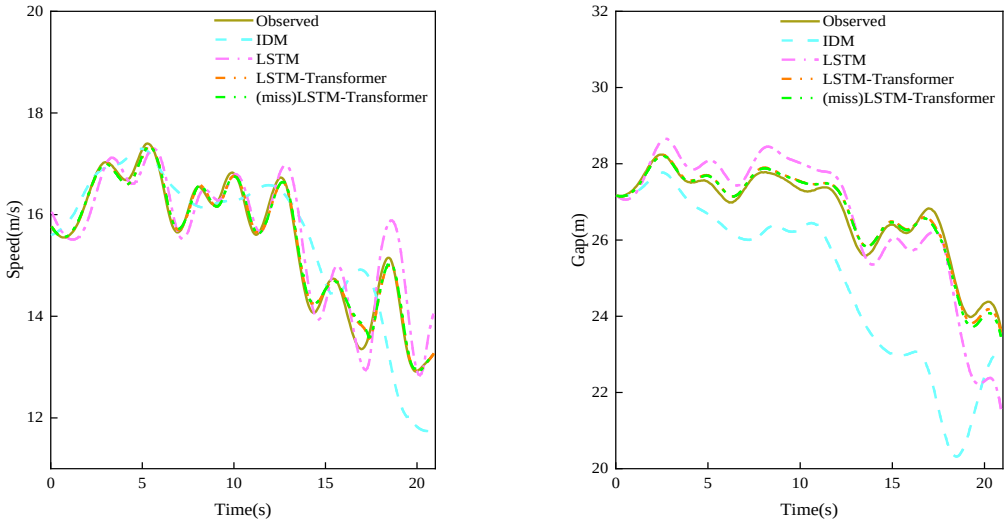


Figure 6. Performance comparison of models.

To verify the performance of different car-following models, we use the test dataset (210 car-following segments) to simulate the three models. The error statistics of other models are shown in Table 5 (‘miss’ represents the performance of the model when the data is missing). For the test results, the error distribution range and the mean value of IDM is more significant, indicating that the LSTM-Transformer and LSTM models have more stable performance than IDM. Although there are differences in the road environment and traffic conditions, in the case of using the natural driving dataset, the LSTM-Transformer model reduces the average MSE of speed simulation by 76.3% and 59.9% compared with IDM and LSTM models, respectively, when the features are not lost, and the average MSE of speed simulation by 72.5% and 53.3%, respectively, when the features are lost. In the case of using NGSIM dataset, the LSTM-Transformer model

reduces the average MSE of speed simulation by 59.2% and 52.3% compared with IDM and LSTM, respectively, when the features are not lost, and the average MSE of speed simulation by 57.1% and 49.8%, respectively, when the features are lost, indicating that it has high accuracy and strong reconstruction ability. We randomly select a car-following segment from the test results to further analyze the differences between the three models, as shown in Figure 6. When the features are not lost or lost, the speed simulation trajectory generated by the LSTM-Transformer model is the closest to the actual trajectory, indicating that the model can effectively extract the feature change information of the car-following data.

3.3. The ability of the model to reproduce traffic phenomena

When the model's accuracy is improved, it can simulate the traffic phenomenon more accurately, and its reproducibility is also enhanced, which can better help us understand and predict the traffic phenomenon. Therefore, to verify the reproducibility of the LSTM-Transformer model when the car-following features are not lost and lost, it is used to reproduce some common traffic phenomena, such as asymmetric driving behavior, traffic oscillation and lag.

3.3.1. Asymmetric driving behavior

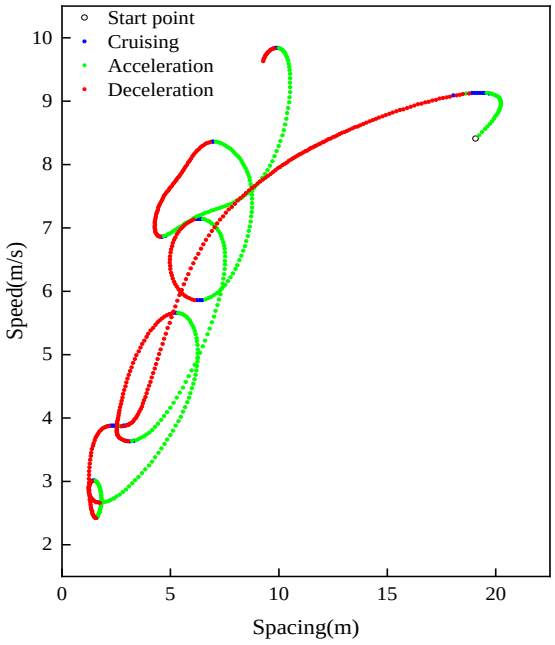
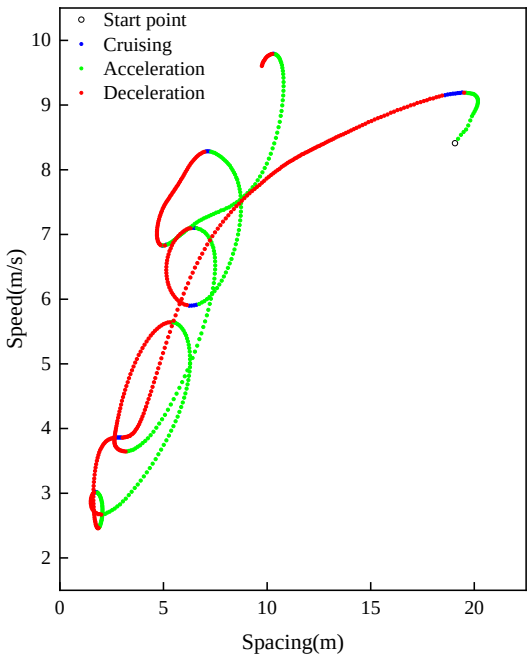


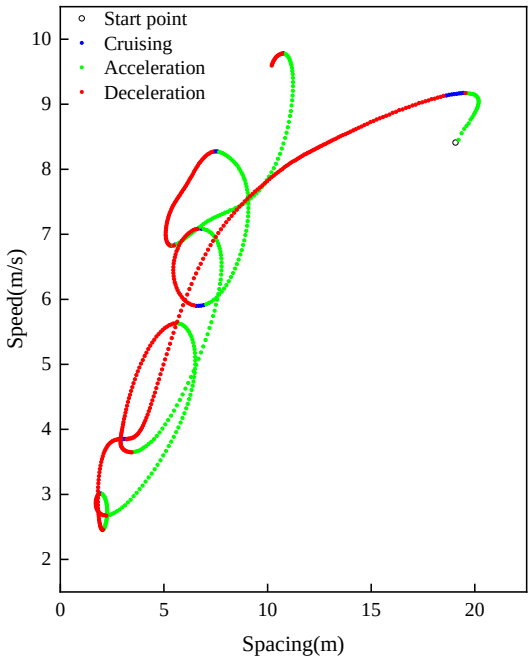
Figure 7. Spacing-speed diagram of observation trajectory.

In most traffic flow models, it is usually assumed that the acceleration and deceleration processes are symmetrical, and it is not realistic to believe that the driver will accelerate or decelerate at the same rate for a specific stimulus. Forbes [25] found that the driver's acceleration response is slower than the deceleration response. Suppose the guided vehicle accelerates immediately after a sudden deceleration. In that case, the headway maintained by the driver of the following car will be twice as long as the headway maintained before deceleration. Driving behavior under different traffic flow conditions is very different (i.e., asymmetric driving behavior), and hysteresis is an essential

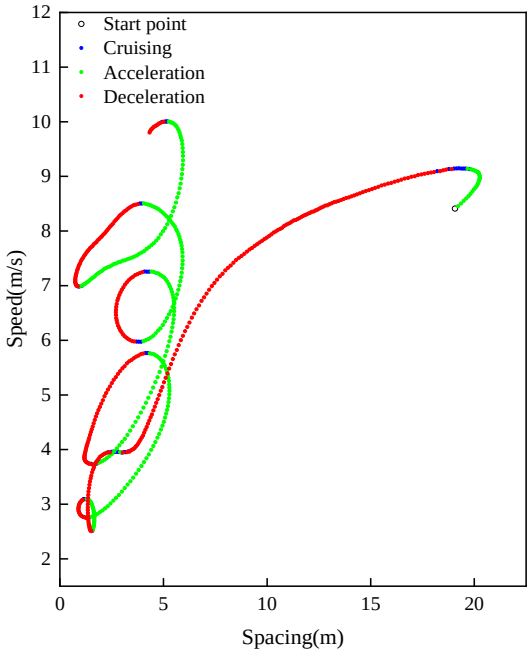
indicator of asymmetric driving behavior [26,27]. We randomly select a car-following segment from the test set to draw the spacing-speed diagram in Figure 7, showing a clear hysteresis loop. To explain this phenomenon, Newell [28] proposed a theory to explain the asymmetry of acceleration and deceleration, which holds that the following car will lag for a while before deciding to respond accordingly when the guiding car accelerates.



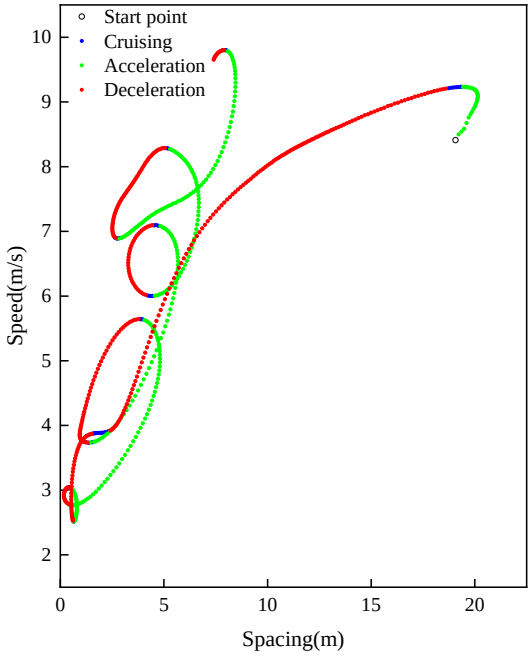
(a) LSTM-transformer model



(b) (miss)LSTM-transformer model



(c) IDM model



(d) LSTM model

Figure 8. Spacing-speed diagram of different models.

The three simulated trajectories reproduced by the LSTM-Transformer, IDM and LSTM models correspond to the observed trajectories in Figure 7, respectively, as shown in Figure 8. Compared with Figure 8(c),(d), the simulated trajectories reproduced in Figure 8(a),(b) are closer to the observed trajectories, indicating that the LSTM-Transformer model has a solid ability to describe asymmetric driving behavior when the features are not lost and lost and can capture driving features from real traffic.

3.3.2. Traffic oscillation and lag phenomenon

In Section 3.3.1, the ability of the LSTM-Transformer model to describe asymmetric driving behavior is evaluated. To further verify the model's performance in reproducing traffic phenomena such as oscillation and lag, a platoon with traffic waves is used for platoon simulation, as shown in Figure 9(a). The average MSE values of the platoon simulation when the features are not lost and lost are 8.038 and 12.011, respectively. The three trajectories corresponding to the simulated platoon and the observation platoon consistently show that the traffic waves propagate propagate upstream, showing typical traffic oscillations, indicating that the model can capture actual traffic characteristics.

Traffic lag is the delay phenomenon of speed recovery when the vehicle emerges from the kinematic disturbance in traffic engineering, manifested by the different speed-density and flow-density curves of accelerating and decelerating traffic flow. Edie [29] collected traffic flow data by aerial survey and obtained the speed-density relationship of a fleet of 13 vehicles within 30 s. The acceleration and deceleration curves are separated, and an acceleration and deceleration behavior pair forms a lag loop. Treiterer and Myers [30] proposed a lag loop on the speed-density plane by using the vehicle trajectory data extracted from aerial maps.

To analyze the hysteresis phenomenon, we use Laval's aggregation method [31] to calculate the traffic flow and density through the parallelogram group in the space-time diagram and draw the corresponding line chart to represent the hysteresis loop. In Figure 9(a), a black parallelogram, the slope of the long side represents the traffic wave's propagation speed, and the short side's slope represents the vehicle's speed. A single parallelogram area's traffic flow and density are calculated according to Eqs (23)–(25).

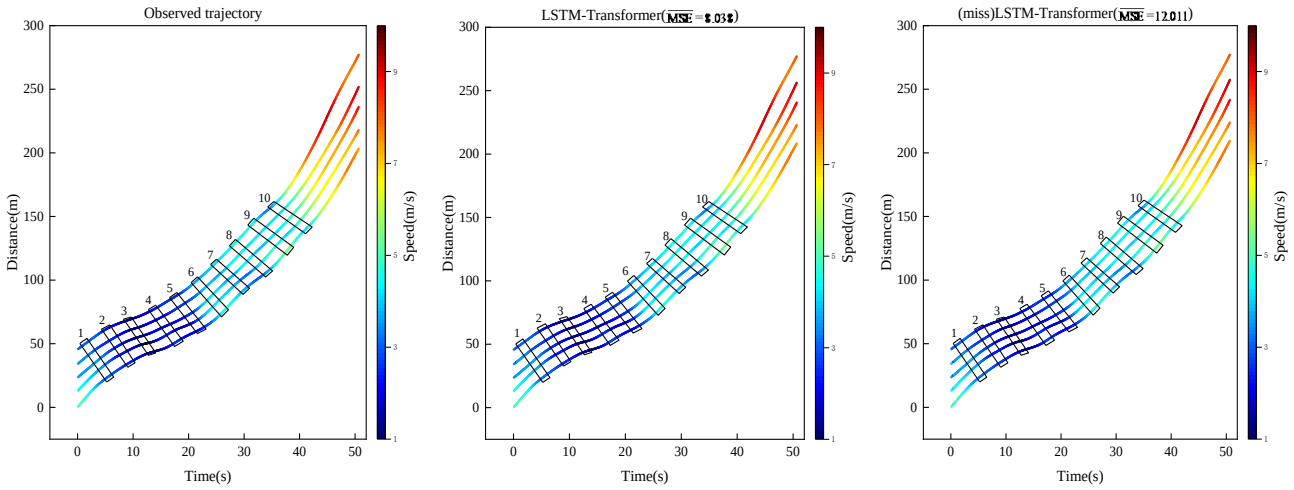
$$k = \sum_{i=1}^{n} t_i \Big/ |\mathrm{A}| \quad , \tag{23}$$

$$q = \sum_{i=1}^{n} x_i \Big/ |\mathrm{A}| \quad , \tag{24}$$

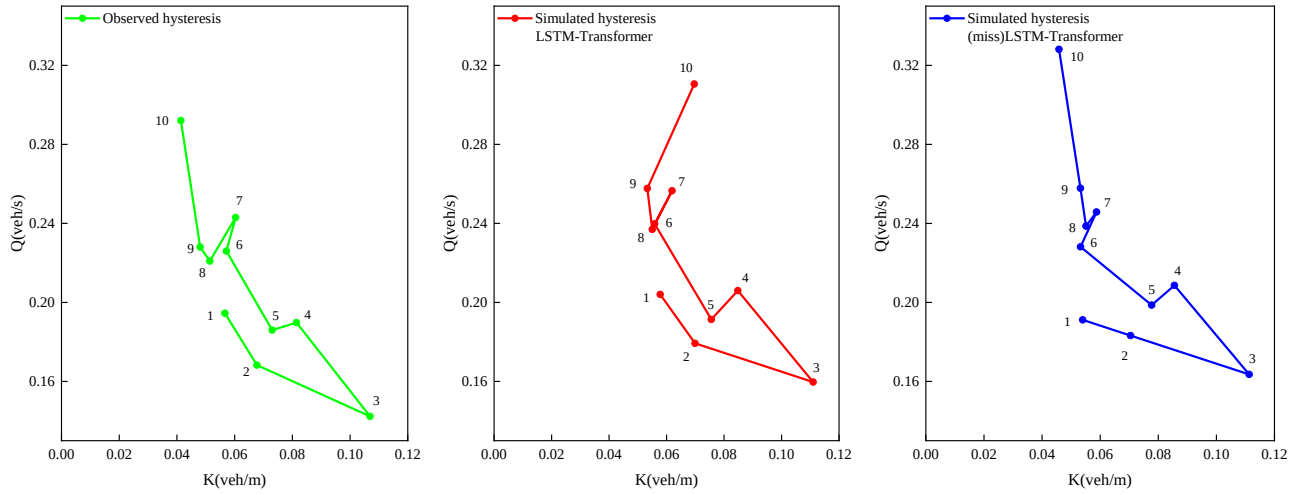
$$v = q/k = \sum_{i=1}^{n} x_i \Big/ \sum_{i=1}^{n} t_i \quad , \tag{25}$$

where k and q represent the density and flow of a single parallelogram area A. v is the speed. $|\mathrm{A}|$ is the area of A. t_i and x_i represent the driving time and distance of the ith vehicle in area A, respectively.

The calculated traffic density diagram is shown in Figure 9(b), which shows that the lag phenomenon generated by the LSTM-Transformer model is in line with the actual situation when the features are not lost and lost, indicating that the model can accurately extract the characteristics of the platoon to reproduce the complex traffic phenomenon.



(a) Space-time diagram of observation platoon and simulation platoon



(b) Flow density diagram

Figure 9. The comparison between the observation platoon and simulation platoon.

4. Conclusions

In this paper, a car-following model based on LSTM-Transformer is proposed. The LSTM unit is the neuron of the transformer used to reconstruct the lost features. The natural driving and NGSIM datasets are used as training and test data and compared with the IDM and LSTM models. By analyzing the mean square error distribution of the reconstructed features from the LSTM-Transformer model, it can be concluded that the model has stable reconstruction quality. The LSTM-Transformer model can reduce the mean square error of speed simulation by 59.9%, and the mean square error can still be reduced by 53.3% in the case of feature loss, indicating that the model can effectively extract the car-following features in the case of feature loss. In addition, the LSTM-Transformer model can accurately reproduce traffic phenomena such as asymmetric driving behavior, traffic oscillation and lag when the car-following features are not lost or lost, indicating that the model not only can capture complex driving behaviors by learning basic information from

actual traffic data but also has strong model learning ability and high reproduction accuracy.

The simulated trajectory of the LSTM-Transformer model is closest to the observed trajectory, indicating that the model can effectively simulate human driving behavior. This study only considers the loss of continuity of car-following features caused by sensors and does not consider various types of feature loss, such as discontinuity and intermittence. Future research should combine more feature loss situations for model testing and verification to adapt to multiple scenarios, thereby expanding the model's adaptability.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by the Guangxi Science and Technology Major Project Funding (Task Book No.: GuiKe AA23062001), the Guangxi Natural Science Foundation Project (grant number: 2019JJA160121) and the Guangxi Science and Technology Base and Talent Project: Construction of Guangxi Transportation New Technology Transfer Center Platform (Task Book No.: Guike AD23026029).

Conflict of interest

The authors declare there is no conflict of interest.

References

1. V. Papathanasopoulou, C. Antoniou, Towards data-driven car-following models, *Transp. Res. Part C Emerging Technol.*, **55** (2015), 496–509. https://doi.org/10.1016/j.trc.2015.02.016
2. M. Saifuzzaman, Z. Zheng, Incorporating human-factors in car-following models: A review of recent developments and research needs, *Transp. Res. Part C Emerging Technol.*, **48** (2014), 379–403. https://doi.org/10.1016/j.trc.2014.09.008
3. V. Punzo, Z. Zheng, M. Montanino, About calibration of car-following dynamics of automated and human-driven vehicles: Methodology, guidelines and codes, *Transp. Res. Part C Emerging Technol.*, **128** (2021), 103165. https://doi.org/10.1016/j.trc.2021.103165
4. Z. Mo, R. Shi, X. Di, A physics-informed deep learning paradigm for car-following models, *Transp. Res. Part C Emerging Technol.*, **130** (2021), 103240. https://doi.org/10.1016/j.trc.2021.103240
5. J. Liu, R. Jiang, J. Zhao, W. Shen, A quantile-regression physics-informed deep learning for car-following model, *Transp. Res. Part C Emerging Technol.*, **154** (2023), 104275. https://doi.org/10.1016/j.trc.2023.104275
6. Z. Mo, X. Di, Uncertainty quantification of car-following behaviors: physics-informed generative adversarial networks, the 28th ACM SIGKDD in conjunction with the 11th International Workshop on Urban Computing (UrbComp2022), 2022. Available from: http://urban-computing.com/urbcomp2022/file/UrbComp2022_paper_3574.

7. M. Montanino, V. Punzo, Trajectory data reconstruction and simulation-based validation against macroscopic traffic patterns, *Transp. Res. Part B Methodol.*, **80** (2015), 82–106. https://doi.org/10.1016/j.trb.2015.06.010
8. X. Wang, Y. Ge, L. Niu, Y. He, T. Z. Qiu, Method for imputing missing data using online calibration for urban freeway control, *Transp. Res. Rec.*, **2672** (2018), 44–54. https://doi.org/10.1177/0361198118774735
9. S. Tak, S. Woo, H. Yeo, Data-driven imputation method for traffic data in sectional units of road links, *IEEE Trans. Intell. Transp. Syst.*, **17** (2016), 1762–1771. https://doi.org/10.1109/TITS.2016.2530312
10. J. M. Chiou, Y. C. Zhang, W. H. Chen, C. W. Chang, A functional data approach to missing value imputation and outlier detection for traffic flow data, *Transportmetrica B: Transport Dyn.*, **2** (2014), 106–129. https://doi.org/10.1080/21680566.2014.892847
11. J. Tang, Y. Wang, S. Zhang, H. Wang, F. Liu, S. Yu, On missing traffic data imputation based on fuzzy C-means method by considering spatial–temporal correlation, *Transp. Res. Rec.*, **2528** (2019), 86–95. https://doi.org/10.3141/2528-10
12. J. Zhao, Y. Gao, J. Tang, L. Zhu, J. Ma, Highway travel time prediction using sparse tensor completion tactics and K-Nearest neighbor pattern matching method, *J. Adv. Transp.*, **2018** (2018), 1–16. https://doi.org/10.1155/2018/5721058
13. Y. Duan, Y. Lv, Y. L. Liu, F. Y. Wang, An efficient realization of deep learning for traffic data imputation, *Transp. Res. Part C Emerging Technol.*, **72** (2016), 168–181. https://doi.org/10.1016/j.trc.2016.09.015
14. Y. Zhuang, R. Ke, Y. Wang, Innovative method for traffic data imputation based on convolutional neural network, *IET Intell. Transp. Syst.*, **13** (2018), 605–613. https://doi.org/10.1049/iet-its.2018.5114
15. D. Zhao, Y. Zhang, W. Wang, X. Hua, M. Yang, Car-following trajectory data imputation with adversarial convolutional neural network, *IET Intell. Transp. Syst.*, **17** (2022), 960–972. https://doi.org/10.1049/itr2.12319
16. Y. Liang, Z. Zhao, L. Sun, Memory-augmented dynamic graph convolution networks for traffic data imputation with diverse missing patterns, *Transp. Res. Part C Emerging Technol.*, **143** (2022), 103826. https://doi.org/10.1016/j.trc.2022.103826
17. C. Zhao, A. Song, Y. Du, B. Yang, TrajGAT: A map-embedded graph attention network for real-time vehicle trajectory imputation of roadside perception, *Transp. Res. Part C Emerging Technol.*, **142** (2022), 103787. https://doi.org/10.1016/j.trc.2022.103787
18. Q. Wan, G. Peng, Z. Li, F. H. T. Inomata, Spatiotemporal trajectory characteristic analysis for traffic state transition prediction near expressway merge bottleneck, *Transp. Res. Part C Emerging Technol.*, **117** (2020), 102682. https://doi.org/10.1016/j.trc.2020.102682
19. A. Sherstinsky, Fundamentals of Recurrent Neural Network (RNN) and Long Short-TermMemory (LSTM) network, *Physica D*, **404** (2020), 132306. https://doi.org/10.1016/j.physd.2019.132306
20. S. Dong, P. Wang, K. Abbas, A survey on deep learning and its applications, *Comput. Sci. Rev.*, **40** (2021), 100379. https://doi.org/10.1016/j.cosrev.2021.100379
21. C. Sun, J. Leng, F. Sun, A fast optimal speed planning system in arterial roads for intelligent and connected vehicles, *IEEE Internet Things J.*, **9** (2022), 20295–20307. https://doi.org/10.1109/JIOT.2022.3172009

22. P. Qin, H. Li, Z. Li, W. Guan, Y. He, A CNN-LSTM car-following model considering generalization ability, *Sensors*, **23** (2023), 660. https://doi.org/10.3390/s23020660
23. L. Ma, S. Qu, J. Ren, X. Zhang, Mixed traffic flow of human-driven vehicles and connected autonomous vehicles: String stability and fundamental diagram, *Math. Biosci. Eng.*, **20** (2022), 2280–2295. https://doi.org/10.3934/mbe.2023107
24. L. Qu, L. Li, Y. Zhang, J. Hu, PPCA-based missing data imputation for traffic flow volume: A systematical approach, *IEEE Trans. Intell. Transp. Syst.*, **10** (2009), 512–522. https://doi.org/10.1109/TITS.2009.2026312
25. T. W. Forbes, Human factor considerations in traffic flow theory, *Highway Res. Rec.*, (1963), 60–66.
26. H. Gong, H. Liu, B. H. Wang, An asymmetric full velocity difference car-following model, *Physica A*, **387** (2008), 2595–2602. https://doi.org/10.1016/j.physa.2008.01.038
27. D. Wei, H. Liu, Analysis of asymmetric driving behavior using a self-learning approach, *Transp. Res. Part B Methodol.*, **47** (2013), 1–14. https://doi.org/10.1016/j.trb.2012.09.003
28. G. F. Newell, Instability in dense highway traffic: A review, *Highway Res. Rec.*, (1965), 73–83.
29. L. C. Edie, *Discussion of Traffic Stream Measurements and Definitions*, Port of New York Authority, (1965), 139–154.
30. J. Treiterer, J. Myers, The hysteresis phenomenon in traffic flow, *Transp. Traffic Theory*, **6** (1974), 13–38.
31. J. A. Laval, Hysteresis in traffic flow revisited: An improved measurement method, *Transp. Res. Part B Methodol.*, **45** (2011), 385–391. https://doi.org/10.1016/j.trb.2010.07.006

AIMS Press