Chaos
An Interdisciplinary Journal of Nonlinear Science

RESEARCH ARTICLE | FEBRUARY 26 2020

A noise-immune LSTM network for short-term traffic flow forecasting

Lingru Cai ; Mingqin Lei ; Shuangyi Zhang ; Yidan Yu ; Teng Zhou ; Jing Qin

Chaos 30, 023135 (2020)
https://doi.org/10.1063/1.5120502

A noise-immune LSTM network for short-term traffic flow forecasting

Cite as: Chaos **30**, 023135 (2020); doi: 10.1063/1.5120502
Submitted: 21 July 2019 · Accepted: 10 February 2020 ·
Published Online: 26 February 2020

Lingru Cai,[1,2] Mingqin Lei,[1] Shuangyi Zhang,[1] Yidan Yu,[1] Teng Zhou,[1,2,3,a)] and Jing Qin[3]

AFFILIATIONS

[1]Department of Computer Science, College of Engineering, Shantou University, 515063 Shantou, China
[2]Key Laboratory of Intelligent Manufacturing Technology, Shantou University, Ministry of Education, 515063 Shantou, China
[3]Center for Smart Health, School of Nursing, The Hong Kong Polytechnic University, 999077 Hong Kong, China

[a)]**Author to whom correspondence should be addressed:** zhouteng@stu.edu.cn

ABSTRACT

Accurate and timely short-term traffic flow forecasting plays a key role in intelligent transportation systems, especially for prospective traffic control. For the past decade, a series of methods have been developed for short-term traffic flow forecasting. However, due to the intrinsic stochastic and evolutionary trend, accurate forecasting remains challenging. In this paper, we propose a noise-immune long short-term memory (NiLSTM) network for short-term traffic flow forecasting, which embeds a noise-immune loss function deduced by maximum correntropy into the long short-term memory (LSTM) network. Different from the conventional LSTM network equipped with the mean square error loss, the maximum correntropy induced loss is a local similar metric, which is immunized to non-Gaussian noises. Extensive experiments on four benchmark datasets demonstrate the superior performance of our NiLSTM network by comparing it with the frequently used models and state-of-the-art models.

Traffic flow modeling is a key component of modern intelligent transportation systems that is of critical importance for proactive traffic management and control systems. Accurate traffic flow modeling can not only subsequently help to alleviate traffic congestion and reduce carbon emissions, but also ensures the efficiency of traffic operation. Traffic flow models can be grouped into microscopic and macroscopic ones. Microscopic models consider the individual behavior and the relationship of the motion for each vehicle, whereas macroscopic models focus on the global properties of traffic flow, such as traffic flow rate (vehicles pass a point per hour) and traffic density (vehicles on the road per kilometer). In this paper, we investigate the traffic flow rate, hereinafter referred to as traffic flow, by macroscopically modeling. We find that the non-Gaussian noises inside the traffic flow data degrade the performance of the long short-term memory network and propose a noise-immune long short-term memory network for short-term traffic flow forecasting. The empirical study confirms our findings and shows the superior performance of our model.

I. INTRODUCTION

Short-term traffic flow forecasting gains increasing attention due to its wide applications in intelligent transportation systems,[1] which is one of the key techniques for traffic control systems,[2] traveler information systems,[3] and vehicle navigation systems.[4] However, accurate traffic flow forecasting remains challenging, since traffic flow naturally contains uncertainty caused by inner and interstochastic dynamics.[5–7]

A series of parametric and nonparametric short-term traffic flow forecasting approaches have been developed in the literature.[8] Historical average,[9] exponential smoothing,[10,11] Kalman filter,[12–14] auto-regressive integrated moving average (ARIMA),[15,16] seasonal auto-regressive integrated moving average (SARIMA),[17] spectral analysis,[18] and structural time-series[19] are categorized as parametric approaches. The parametric approaches achieve good performances when expertise domain knowledge is fully infused to tune optimal parameters for such models. Other researchers pay attention to nonparametric approaches, such as artificial neural network (ANN),[20,21] extreme learning machine (ELM),[22] k-nearest neighbor (KNN),[23] and support vector machine (SVM).[24–26]

Recently, deep learning techniques have gained great achievements in various domains.[27] Compared with conventional shallow learning algorithms, deep neural networks can model complex non-linear relationships by distributed and hierarchical feature representation.[28–30] Lv *et al.*[31] pioneered a deep architecture model called stacked autoencoder (SAE) for traffic flow prediction.

Zhou *et al.*[29] found that a single SAE with fixed parameters can hardly handle various traffic conditions, and they proposed a δ-agree AdaBoost strategy to integrate a series of stacked autoencoders for better forecasting according to this finding. Then, Zhou *et al.*[30] also proposed a deep learning framework to integrate heterogeneous forecasting models. The recurrent neural network (RNN) has been proven to be effective for traffic flow forecasting,[32] which is initially used for machine translation, and later transferred to temporal-spatial tasks, such as traffic flow forecasting. However, the conventional recurrent neural network suffers from gradient explosion and gradient vanishing.[33] To achieve the long-term dependencies, a special recurrent neural network, termed long short-term memory (LSTM),[34,35] is developed to capture temporal features in a long period. Ma *et al.*[36] applied the LSTM for traffic speed prediction from remote microwave sensor data. Yongxue and Li[37] used LSTM for traffic prediction and claimed that LSTM outperforms most of the other nonparametric models.

For the typical setting of LSTM networks, the mean square error (MSE) is the most widely used cost function due to its attractive features, such as smoothness, convexity, and low computational burden under the Gaussian assumption. However, the successful deployment of such LSTM networks heavily rely on the Gaussianity and linearity assumptions, since the MSE loss embedded in a conventional LSTM network aims to measure the overall similarity of two random vectors, which is optimal for the case of independent and identically distribution Gaussian noise, but not robust to non-Gaussian noises.[38] However, these assumptions do not always hold for the application of traffic flow forecasting, because the non-Gaussian noises inside the traffic flow data may be caused by hardware failure, manual traffic control, or unexpected accidents, etc.[13] Thus, designing a proper cost function is vital for the traffic flow forecasting task. Liu *et al.*[39] extended the concept of the correntropy criterion from the information learning theory for general similarity measurement between two random vectors with non-Gaussian and impulsive noises. The correntropy criterion is a robust metric under the non-Gaussian noise assumption, which has been successfully applied to face image recognition,[40] wind power forecasting,[41] principal component analysis,[42] subspace clustering,[38] and regression problem.[43] Based on the correntropy criterion, a local loss metric can be further deduced. When the errors are relatively small, the loss is close to the absolute loss. For large losses that are usually caused by non-Gaussian noises, the loss is close to 1. Thus, the influence of the non-Gaussian noises is eliminated by this loss function.

To achieve accurate traffic flow forecasting under the real-world situation, where the traffic flow is often mixed with non-Gaussian noises, we propose a noise-immune LSTM network by employing the correntropy criterion. The main contributions of this paper are as follows.

- First, we explore the inner regular pattern of the traffic flow data by setting different input lengths. By selecting a certain range of the input length, we reconstruct a training dataset.
- Then, we propose a noise-immune LSTM network, named NiLSTM, that equipped with the noise-immune loss deduced by the correntropy criterion to eliminate the effect of the non-Gaussian noises inside the traffic flow.
- Third, we evaluate our network on four benchmark datasets and compare it with several state-of-the-art models by an empirical study. The results show that our NiLSTM achieves improvement on the four datasets than the conventional LSTM model and other control models.
- Besides, we also analyze the robustness of our NiLSTM model by exploring different settings of the kernel size.

II. METHODOLOGY

In this section, we will describe the LSTM network for traffic flow forecasting. Then, we propose a correntropy induced loss for the LSTM network for traffic flow forecasting.

A. LSTM network for traffic flow forecasting

The LSTM network[34,35] is a special kind of recurrent neural networks (RNNs), which overcomes the gradient vanishing and exploding issues of the conventional RNN.[33] In the LSTM architecture, three peculiar structures endue the capability to handle the correlation within time series in both the short and long terms, which are a forget gate, an input gate, and an output gate, respectively. The forget gate discards information from the cell state. The input gate stores information from outside to update the cell state. The output gate takes all results to calculate and generate output for the LSTM structure.

The conventional structure of the LSTM architecture is shown in Fig. 1. Two symbols wrapped in a circle named σ and $\otimes$ denote the standard logistic sigmoid function and matrix multiplication, respectively, while the symbol tanh wrapped by ellipses means the tanh function. Three dotted lines pointed out by C_{t-1} represent the transition of hidden state h_{t-1}, while the other three solid lines pointed out by C_{t-1} mean a normal state transition. Besides, two dotted lines whose root connected to the symbol $\otimes$ complete the state updating process in neurons.

We denote the input traffic flow sequence as $X = \{x_1, x_2, \ldots, x_n\}$, where n is the number of training samples. The hidden state of memory cells is denoted as $h = \{h_1, h_2, \ldots, h_n\}$, and $y = \{y_1, y_2, \ldots, y_n\}$ is the groundtruth. Other symbols in Eqs. (1)–(6) are described as follows, ω denotes weight matrices in hidden layers, and b means the bias vectors. ω_{it}, ω_{ft}, and ω_{ot} represent the weight matrices in the input gate, forget gate, and output gate, respectively. Similarly, b_i, b_f, and b_o correspond to the structure of the three gates.

As shown in Eq. (1), the forget gate takes x_t and h_{t-1} as input and utilizes the sigmoid function to discard information,

$$f_t = \sigma\left(\omega_{ft} x_t + \omega_{ft} h_{t-1} + b_f\right). \tag{1}$$

The input gate is accomplished by Eqs. (2) and (3). First, the sigmoid function decides which value will be updated. Then, the state is added with a candidate vector $\tilde{C}_t$ created by the tanh function,

$$i_t = \sigma\left(\omega_{it} x_t + \omega_{ih} h_{t-1} + b_i\right), \tag{2}$$

$$\tilde{C}_t = \tanh\left(\omega_{ct} h_{t-1} + \omega_{cx} x_t + b_c\right). \tag{3}$$

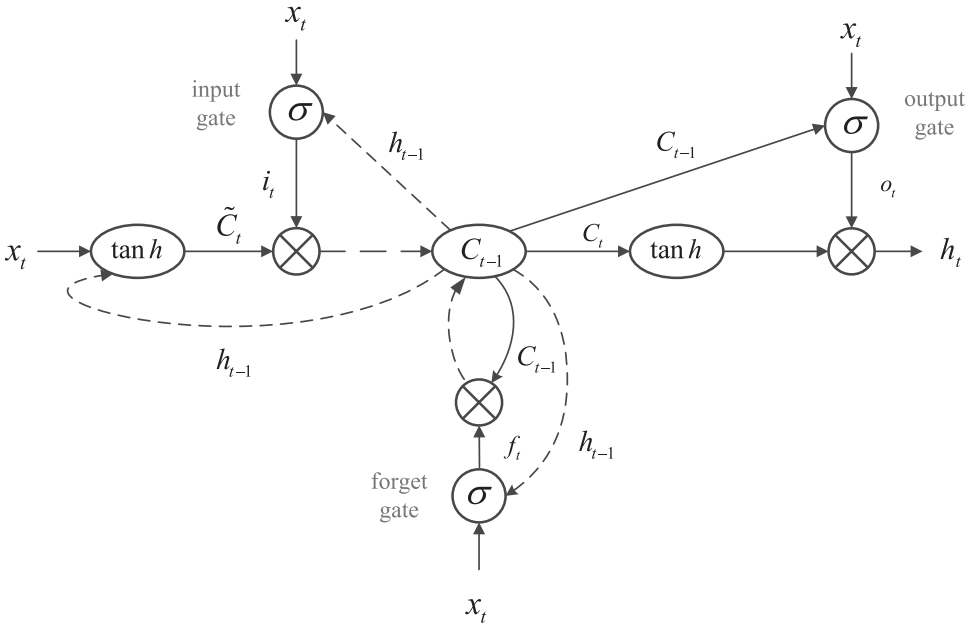

FIG. 1. Structure of a conventional LSTM neural network. The operator $\otimes$ represents matrix multiplication. The operator σ is the logistic sigmoid function.

We update the state of the unit by multiplying the old state C_{t-1} with f_t and add it with $i_t * \tilde{C}_t$, as shown in Eq. (4),

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t. \tag{4}$$

Finally, the output gate produces the final output of h_t. The whole process is divided into two stages described in Eqs. (5) and (6),

$$O_t = \sigma\left(\omega_{ot} x_t + \omega_{oc} C_{t-1} + b_o\right), \tag{5}$$

$$h_t = O_t * \tanh\left(C_t\right), \tag{6}$$

where σ decides the portion of C_{t-1} will be exported. In the end, Eq. (6) combines the new state C_t and the portion to calculate the final output h_t.

B. Noise-immune LSTM network

In conventional LSTM architectures, the mean square error is the most widely used cost function for regression tasks, since it has a few attractive features, such as smoothness, convexity, and low computational burden for data obeying the Gaussian distribution. However, the traffic flow is easily affected by several inner or extra factors, such as manual traffic control or unexpected accidents, which are considered as non-Gaussian noises. The performance of the MSE versions is easily degraded under non-Gaussian situations. Thus, designing a proper cost function for the LSTM network is a successful engineering solution for this task.

Correntropy is a relatively new tool for non-quadratic similar metrics that has been proven to be more robust than MSE under the non-Gaussian situation. Generally, the correntropy is a similar metric between two random variables X and Y,

$$V(X, Y) = \mathbf{E}[\kappa(X, Y)], \tag{7}$$

where $\mathbf{E}(\cdot)$ is the expectation operator and κ is the Gaussian kernel function as Eq. (8),

$$\kappa_\sigma(X, Y) = \exp\left(-\frac{|X - Y|^2}{2\sigma^2}\right), \tag{8}$$

where σ is the kernel size, which is always larger than zero.

Motivated by the successful applications of correntropy, we develop a noise-immune LSTM network by employing such a metric. To calculate the optimal solution by using stochastic gradient descent in the backpropagation process in the LSTM model, we adopt the method developed by Chen *et al.*[44] We multiply the expression of the original correntropy by a factor -1 and add $\kappa(0)$, which is equal to $\kappa_\sigma(0)$ when $X = Y$ regardless of what σ is setting,

$$\kappa(0) - \exp\left(-\frac{|X - Y|^2}{2\sigma^2}\right). \tag{9}$$

For a better interpretation of the correntropy, we make a straightforward comparison between the correntropy and the MSE. Similar to the correntropy, the MSE is defined in Eq. (10),

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(X - Y)^2. \tag{10}$$

In Eq. (10), the MSE takes the square operation of the error, whereas the correntropy in Eq. (8) makes exponential operation. If the error $|X - Y|$ is larger than 1, the square operation will further quadratically increase the MSE. When there are non-Gaussian noises, such as outliers, inside the data, the MSE will magnify the influence of such noises. For the MSE, all samples contribute the same degree impact on the final results, regardless it is a discrete point or not. Different from the MSE, the correntropy is a local metric. Equation (9) shows the following peculiarities.[39] When two

random variables are close, e.g., $\frac{|X-Y|^2}{2\sigma^2}$ is close to 0, the correntropy induced noise-immune loss behaves like the MSE. When two random variables are getting further, e.g., $\frac{|X-Y|^2}{2\sigma^2}$ is less than 1, it does like the mean absolute error (MAE). When the two random variables are far away, e.g., $\frac{|X-Y|^2}{2\sigma^2}$ is larger than 1 and tends to be infinity, the correntropy induced noise-immune loss tends to saturate, e.g., $\kappa(0)$. For the LSTM network, it is common to minimize the loss function by a stochastic gradient descent algorithm to optimize the network. As the conventional MSE loss is sensitive to the non-Gaussian noises, the LSTM network guided by the MSE loss tend to be misled by the non-Gaussian noises inside the traffic flow. Different from the MSE loss, the noise-immune loss is insensitive to non-Gaussian, such as outliers, which leads to large errors. In this regard, the noise-immune loss is sensitive to the error for common cases and insensitive to outliers.

III. EXPERIMENTS

A. Data preparation

The four benchmark datasets were published by Wang *et al.*[45] These traffic flow data were collected by MONICA sensors from four motorways at the time interval of 1 min, namely, A1, A2, A4, and A8 motorways, respectively, in Amsterdam from May 20, 2010, to June 24, 2010. We aggregate these data as vehicles per hour for every 10 min, which coincides with other traffic flow forecasting tasks[12,30,45] on these datasets. The geographical locations of the four measurement points are shown in Fig. 2. We briefly describe these four points on four motorways further.

- As part of the E30 European route, the A1 highway connects Amsterdam to the German border. Meanwhile, as the first high-occupied dual three-lane highway in Europe, whose fluctuations of the traffic flow are more dramatic, making forecasting more difficult.
- The A2 freeway connects the city of Amsterdam and the Belgian border, which is one of the busiest motorways in the Netherlands, with more than 2000 vehicles per hour.
- The A4 motorway starts from Amsterdam and ends at Stabroek, the northern border of Belgium, among 154 km long.
- Starting from the northernmost end of A10 at interchange Knooppunt Coenplein, along the northwest route through Coentunnelweg, the total length of A8 freeway is less than 10 km.

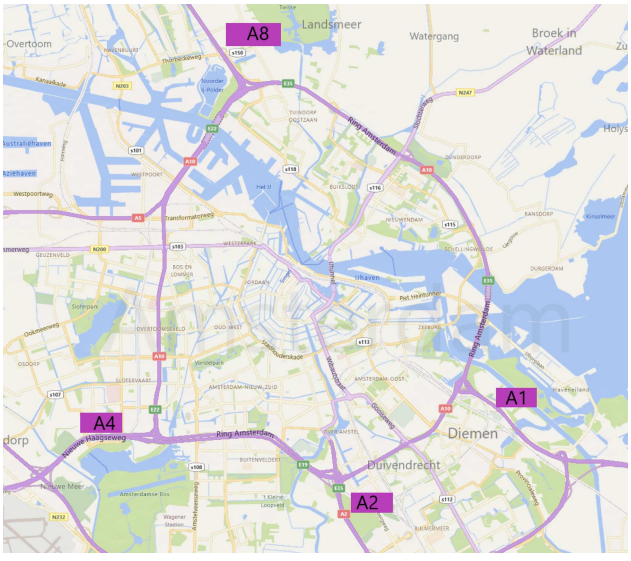

FIG. 2. The measurement points on four motorways of Amsterdam.

B. Evaluation criteria

We employ two commonly used criteria to evaluate the performance of our NiLSTM network, e.g., the root mean square error (RMSE) and the mean absolute percentage error (MAPE). The RMSE measures the average difference between the predictions and the groundtruth. The MAPE accounts for the percentage of such differences. Equations (11) and (12) are the definitions of RMSE and MAPE, respectively,

$$\mathrm{RMSE} = \sqrt{\frac{1}{M}\sum_{m=1}^{M}(v(m)' - v(m))^2}, \tag{11}$$

$$\mathrm{MAPE} = \frac{1}{M}\sum_{m=1}^{M}\left|\frac{v(m)' - v(m)}{v(m)}\right| \times 100\%, \tag{12}$$

where $v(m)'$ and $v(m)$ denote the predictive values and real measurements corresponding to mth group data. Meanwhile, M denotes the total number of samples to be predicted.

C. Experimental setup

Traffic flow is one of the most common traffic parameters, which is the number of vehicles passing a cross section of a roadway in a specified period of time and given in terms of vehicles per hour. Traffic flow forecasting is generally classified into two types based on the length of time ahead to predict, e.g., short-term traffic flow forecasting (5–30 min) and medium-and-long-term traffic flow forecasting (over 30 min).[46] In this study, we set the length of time to 10 min for short-term traffic flow forecasting. The length of the input traffic flow sequence is the amount of information for a model to make forecasting. For instance, the input lengths of the traffic flow sequence are 1, 2, and 3, which means the last 10-min, 20-min, and 30-min of traffic flow sequences are used for the model to make forecasting, respectively. If we would like to forecast the traffic flow x_t at time interval t using the traffic flow of the last 30 min, we set the input traffic flow sequence to $\{x_{t-3}, x_{t-2}, x_{t-1}\}$.

As mentioned above, each dataset contains five weeks of traffic flow sequences. The first four weeks are used for training, while the last week is used for testing. Considering the temporal dependencies of time series data, we utilize a hold-out cross-validation[47] approach to split training data, while the training set is split into the training subset and validation set. More intuitively, the training set, validation set, and test set are divided in a chronological order, where the validation set comes after the training set, and the test set comes after the validation set.

TABLE I. The hyperparameters of the LSTM model.

Hyperparameter	Value
Units	256
Batchsize	32
Epochs	50
Validation_split	0.05

TABLE II. The parameters of the ANN model.

Parameter	Value
Hidden layers	1
Goal	0.001
Spread	2000
MN	40
DF	Default

We employ the hyperas (a toolkit based on Bayesian optimization in the Keras[48] framework) for hyperparameter tuning. The optimized hyperparameters include the units, the batchsize, the epochs, and the validation_split. The units represent the number of hidden neurons in the LSTM model. The batchsize refers to the number of training examples processed in one iteration. In each iteration, a certain number of examples is used to update the parameters of the model by the batch gradient descent algorithm. An epoch indicates

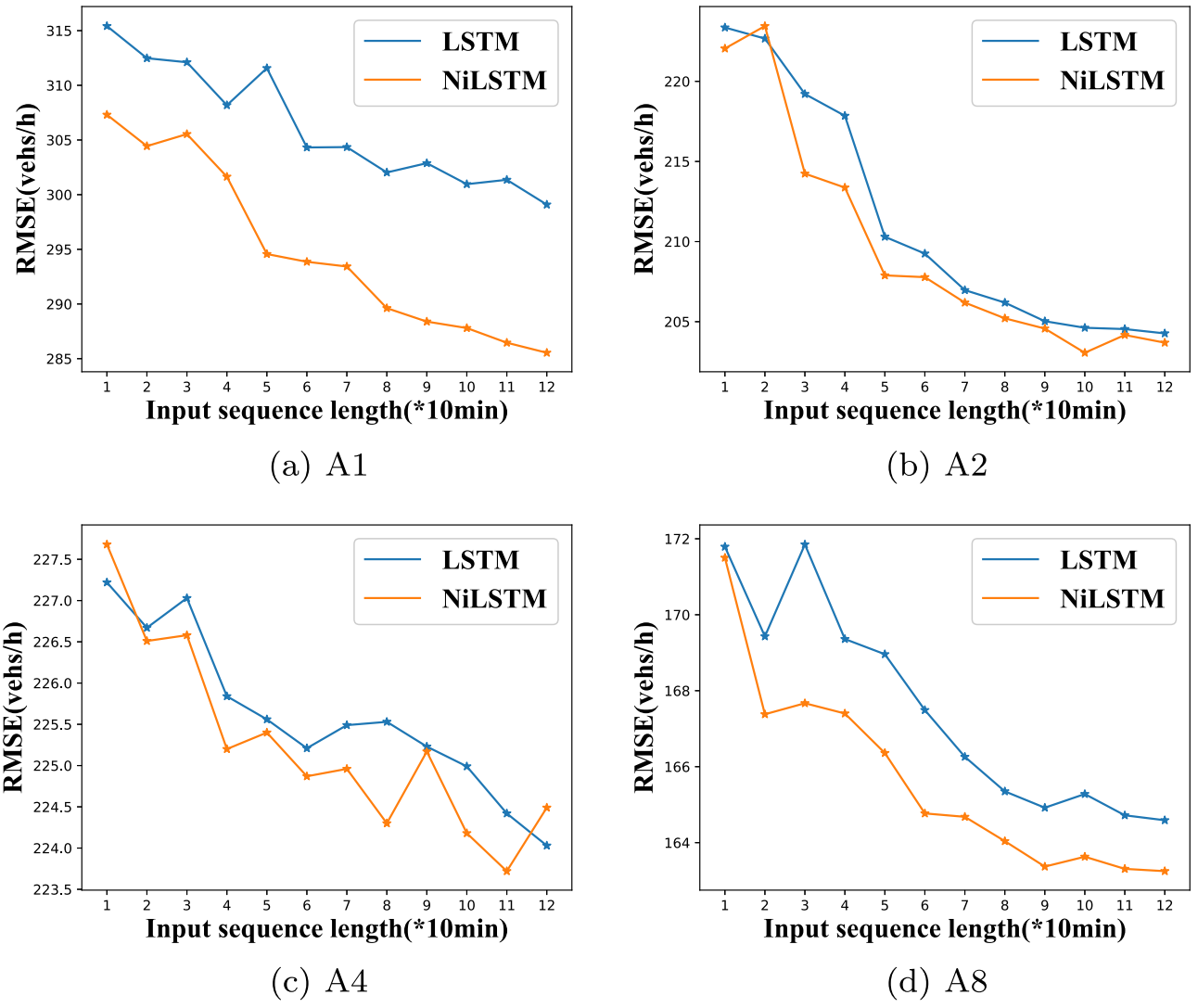

FIG. 3. The RMSEs of the LSTM and the NiLSTM on four datasets.

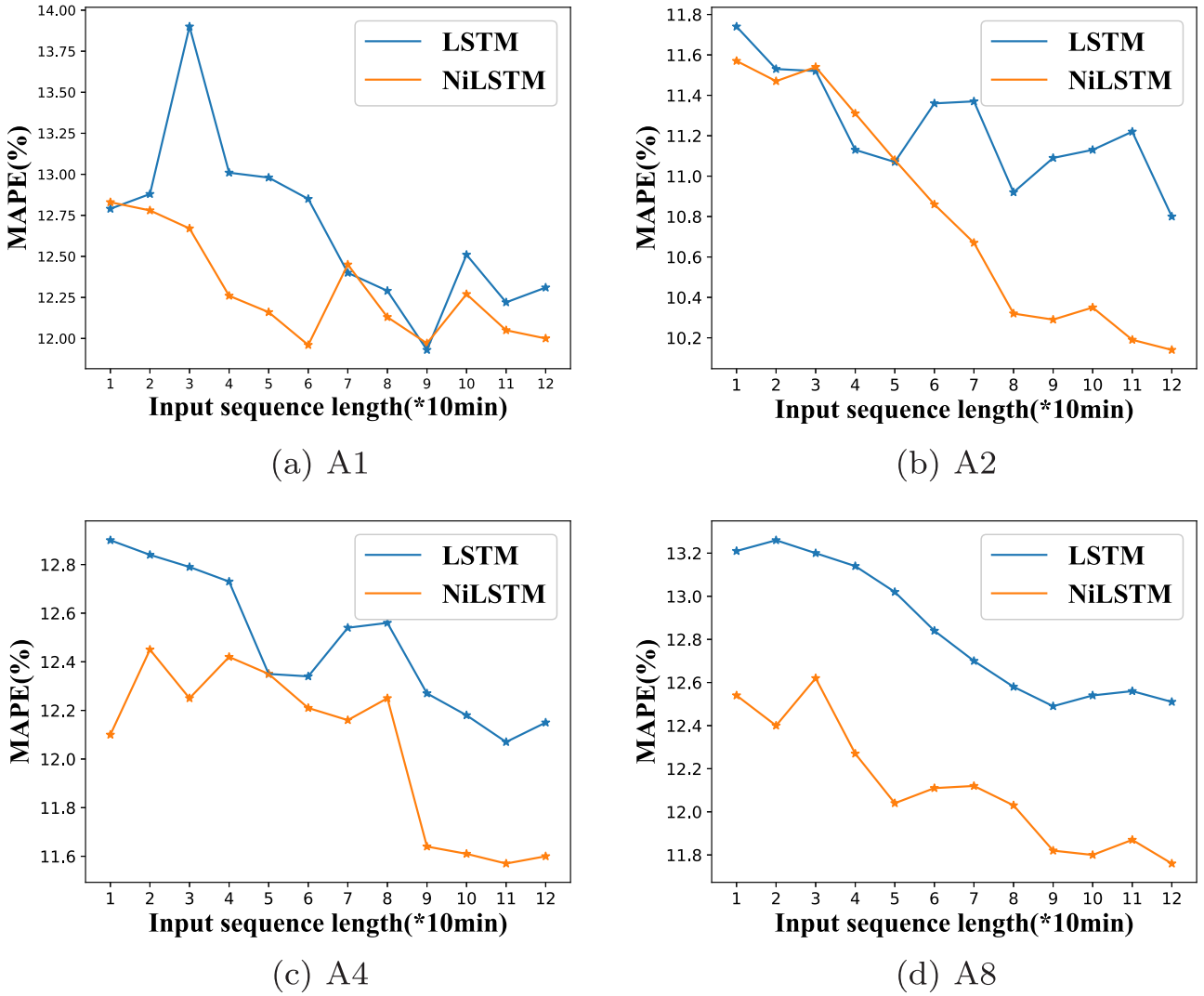

FIG. 4. The MAPEs of the LSTM and the NiLSTM on four datasets.

the entire training set passed forward and backward a deep neural network once. The validation_split controls the portion of the training set used for training and for validation.

The ranges of the unit and batchsize are set to {16, 32, 64, 128, 256}. The range of epochs is set to {10, 20, 30, 40, 50}. The validation_split is set to {0.05, 0.1}. The hyperparameters tuned for the following experiments are listed in Table I.

We also explore the optimal length of the input traffic flow sequence for the conventional LSTM model by adjusting the input length from 1 to 12, e.g., 10 min to 2 h. We conduct the same 12 groups of experiments on our NiLSTM model correspondingly. For each group of length, we repeat the experiment for 20 times and average the results of 20 experiments as the final result to eliminate the randomness of a single experimental result.

We compare our NiLSTM with five frequently used state-of-the-art models, which are autoregression (AR), Kalman filter (KF), artificial neural network (ANN), support vector machine regression (SVR), and stacked auto-encoder (SAE). We also employ two baseline models for traffic flow forecasting, e.g., historical average (HA) and the random walk model (RW). Brief and necessary introductions of these models are as follows.

Historical average (HA): The average traffic flow of a certain time in the past period is used as the forecasting for the current traffic flow.

Random walk (RW): The random walk assumes that, in each period, the traffic flow takes a random step away from its previous value. One step is commonly reported in traffic flow forecasting tasks.[49] More details about HA and RW can be found in Ref. 49.

TABLE III. The forecasting performance of the NiLSTM and other models on four benchmark datasets. The best performances are denoted in boldface.

Model	Criteria	A1	A2	A4	A8
HA	RMSE	404.84	348.96	357.85	218.72
	MAPE	16.87	15.53	16.72	16.24
RW	RMSE	312.92	223.82	230.01	174.14
	MAPE	12.65	11.43	12.07	12.37
AR	RMSE	301.44	214.22	226.12	166.71
	MAPE	13.57	11.59	12.7	12.71
KF	RMSE	332.03	239.87	250.51	187.48
	MAPE	12.46	10.72	12.62	12.63
ANN	RMSE	299.64	212.95	225.86	166.5
	MAPE	12.61	10.89	12.49	12.53
SVR	RMSE	329.09	259.74	253.66	190.3
	MAPE	14.34	12.22	12.23	12.48
SAE	RMSE	295.43	209.32	226.91	167.01
	MAPE	**11.92**	10.23	11.87	12.03
NiLSTM	RMSE	**285.54**	**203.69**	**223.72**	**163.25**
	MAPE	12.00	**10.14**	**11.57**	**11.76**

Autoregressive (AR): This model has been widely used in traffic flow predicting due to the randomness of traffic data. In an autoregressive model with order p, the current traffic flow is represented by weighted combination going back p periods, following a random disturbance in the current period. In this regard, the order p is critical for the model. If the order is too high, more coefficients need to be introduced. The order in our experiment is set to eight by cross-validation.

Kalman filter (KF): A wavelet denoising procedure proposed by Xie *et al.*,[50] which is employed to preprocess the traffic flow data. We use Daubechies 4 as the mother wavelet as suggested in the literature. The variance of the process error Q is set to $0.1 * I$, where I is the identity matrix. The variance of the measurement noise is 0, since we regard that the measurement is correct. The initial state is the set to $[1/n, \ldots, 1/n]$, where n is set as eight. The covariance matrix of the initial state estimation error is set to $10 - 2 * I$.

Artificial neural network (ANN): The ANN is a three-layered feed-forward neural network with one radial basis layer. The adopted parameters are listed in Table II, where the goal is the mean squared error goal, the spread is spread of a radial basis function (RBF), the MN is the maximum number of neurons in a hidden layer, and the DF is the number of neurons to add between displays. More details can be found in Ref. 21.

Support vector machine regression (SVR): For the support vector machine regression model, several parameters need to be set beforehand. The regression horizon is set to eight, which is the same as Zhou *et al.*[29] We use a radial basis function (RBF) as the kernel type in this work. The parameter C is set to the maximum difference between the traffic flow. The width parameter for the RBF kernel is set to 3×10^{-6}.

Stacked autoencoder (SAE): The stacked autoencoder is trained in a layer-wise greedy fashion, see Lv *et al.*[31] The spatial and temporal correlations are inherently considered in this model. The deep architecture of the SAE is set to $[120, 60, 30]$ by cross-validation.

TABLE IV. The performance of the NiLSTM with different kernel sizes.

σ	Criteria	A1	A2	A4	A8
0.1	RMSE	285.73	203.81	224.12	162.84
	MAPE	11.97	10.20	11.68	11.74
0.2	RMSE	285.68	202.84	225.01	162.84
	MAPE	11.95	10.39	11.60	11.77
0.5	RMSE	286.96	203.30	224.50	164.01
	MAPE	11.90	10.48	11.57	11.86
1.0	RMSE	285.54	203.69	224.49	163.25
	MAPE	12.00	10.14	11.60	11.76
2.0	RMSE	285.61	203.50	224.67	164.46
	MAPE	11.80	10.36	11.66	11.86
3.0	RMSE	286.52	204.03	224.53	164.56
	MAPE	12.12	10.54	11.74	11.92

The length of the input sequence for the NiLSTM is set to 12 by cross-validation. Then, we set the range of kernel size to $[0.1, 0.2, 0.5, 1.0, 2.0, 3.0]$ to search a optimal kernel size, which is consistent with Liu *et al.*[39]

D. Results and analysis

In this section, we first evaluate the improvement of the NiLSTM by comparing it with the conventional LSTM by 12 groups of control experiments. Then, we compare the NiLSTM with the baseline and state-of-the-art methods to demonstrate the superiority of the NiLSTM. We also demonstrate the robustness of the NiLSTM within the range of suggested kernel sizes.

1. Comparisons of NiLSTM and LSTM

We evaluate the performance of the LSTM model and the NiLSTM model by 12 groups of control experiments with different lengths of input traffic flow sequences. The results are shown in Figs. 3 and 4, where the horizontal axis is the input length, and the vertical axis denotes the RMSE and MAPE, respectively. There are four figures in Figs. 3 and 4, respectively. Two polylines in the figures account for the RMSE and MAPE of the LSTM and the NiLSTM on A1, A2, A4, and A8 dataset. The blue one represents the RMSE and MAPE of LSTM in Figs. 3 and 4, and the orange one denotes the RMSE and MAPE of the NiLSTM, respectively.

As shown in Fig. 3, the RMSEs of the LSTM and the NiLSTM decreases with the increase of the input length. It indicates the more information is given (within limits) to the models, the better forecasting performance they achieve. The NiLSTM performs better than the LSTM for most cases, especially for A1 and A8. The RMSE is reduced by approximately 11.34 and 1.88, respectively. For these four datasets, it is obvious that the A8 has smaller RMSE, regardless of which method is used. For the inconsistent improvement of the model on different datasets, we assume that it is caused by the intrinsic characteristics of the datasets. Since the other three datasets are coming from the international highways, whereas the A8 highway is a straight road, which all belongs to Amsterdam. A1

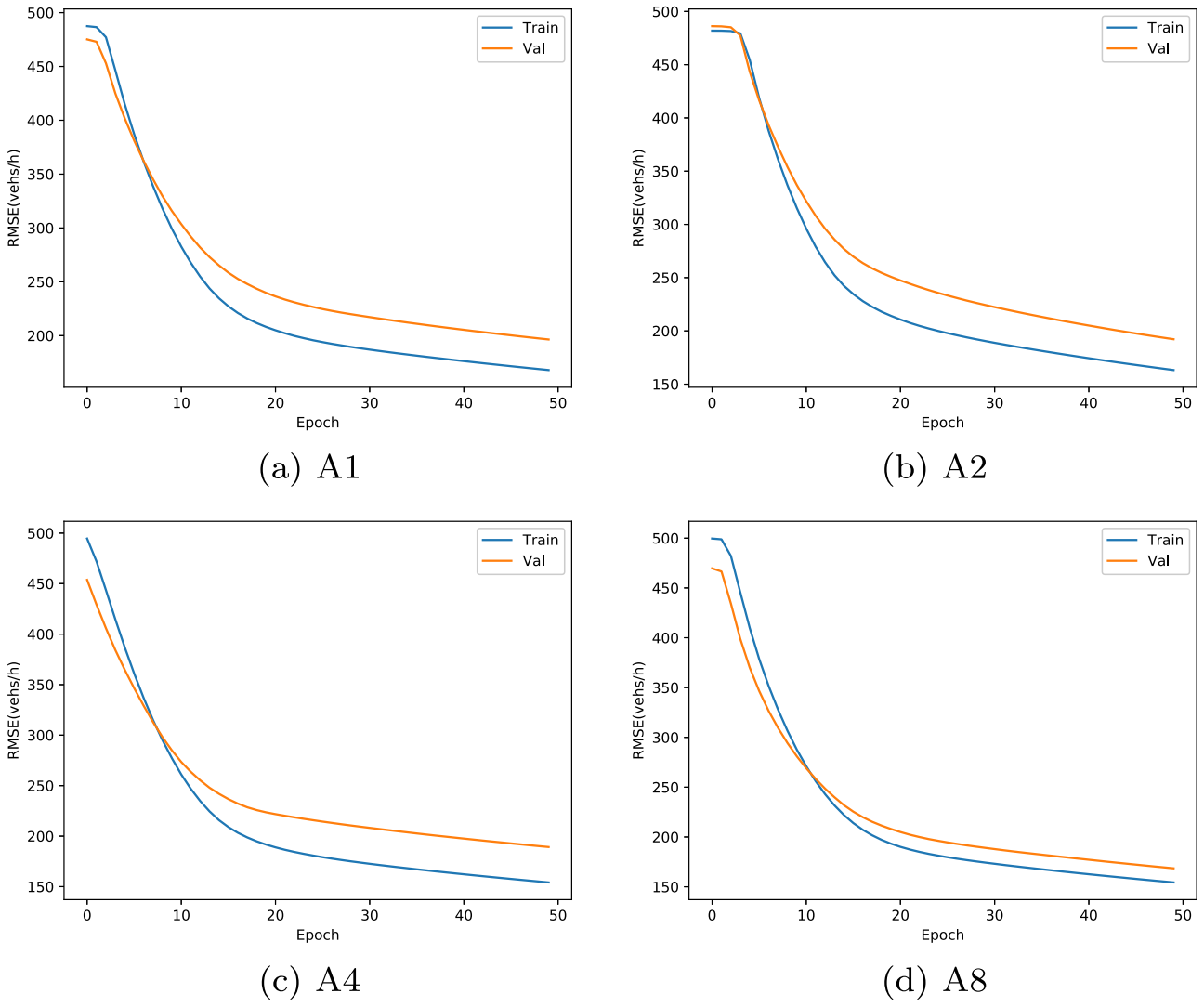

FIG. 5. The RMSE on the training set and the validation set for the training process.

connects to the German border, and A2 and A4 connect to the Belgian border. The traffic flow of the A8 highway does not change so dramatically than A1, A2, and A4, which leads to the lower RMSE of the A8 dataset.

Similarly, we analyze the MAPEs of the LSTM model and the NiLSTM model. For the LSTM, with the increase of input length, the MAPE shows a decreasing trend on the four datasets, in spite of some fluctuations. Comparing the MAPEs for the two models on the four datasets, although there are some intersections in Fig. 4, the NiLSTM achieves better performance than the LSTM as a whole. The NiLSTM achieves great improvement on the A8 dataset regarding the MAPE, and the MAPE is relatively small than the other three datasets. The improvements of the MAPEs on A1, A2, A4, and A8 dataset are 0.38, 0.42, 0.42, and 0.72, respectively.

2. Comparisons with state-of-the-art methods

We also compare the NiLSTM model with seven frequently used state-of-the-art forecasting models. The results are listed in Table III.

In Table III, the parameters of the models are fine-tuned by grid search. For example, the length of the input traffic flow sequences for A1, A2, and A8 are set to 12, whereas the length is set to 11 for A4.

From Table III, we find that the NiLSTM outperforms the conventional parametric and nonparametric methods, obviously. It is because the parametric methods are hard to deal with the nonlinear relationship inside the traffic data with limited parameters and fixed model settings. For example, the HA is easily influenced by unexpected incidents. The KF is prone to overshooting when

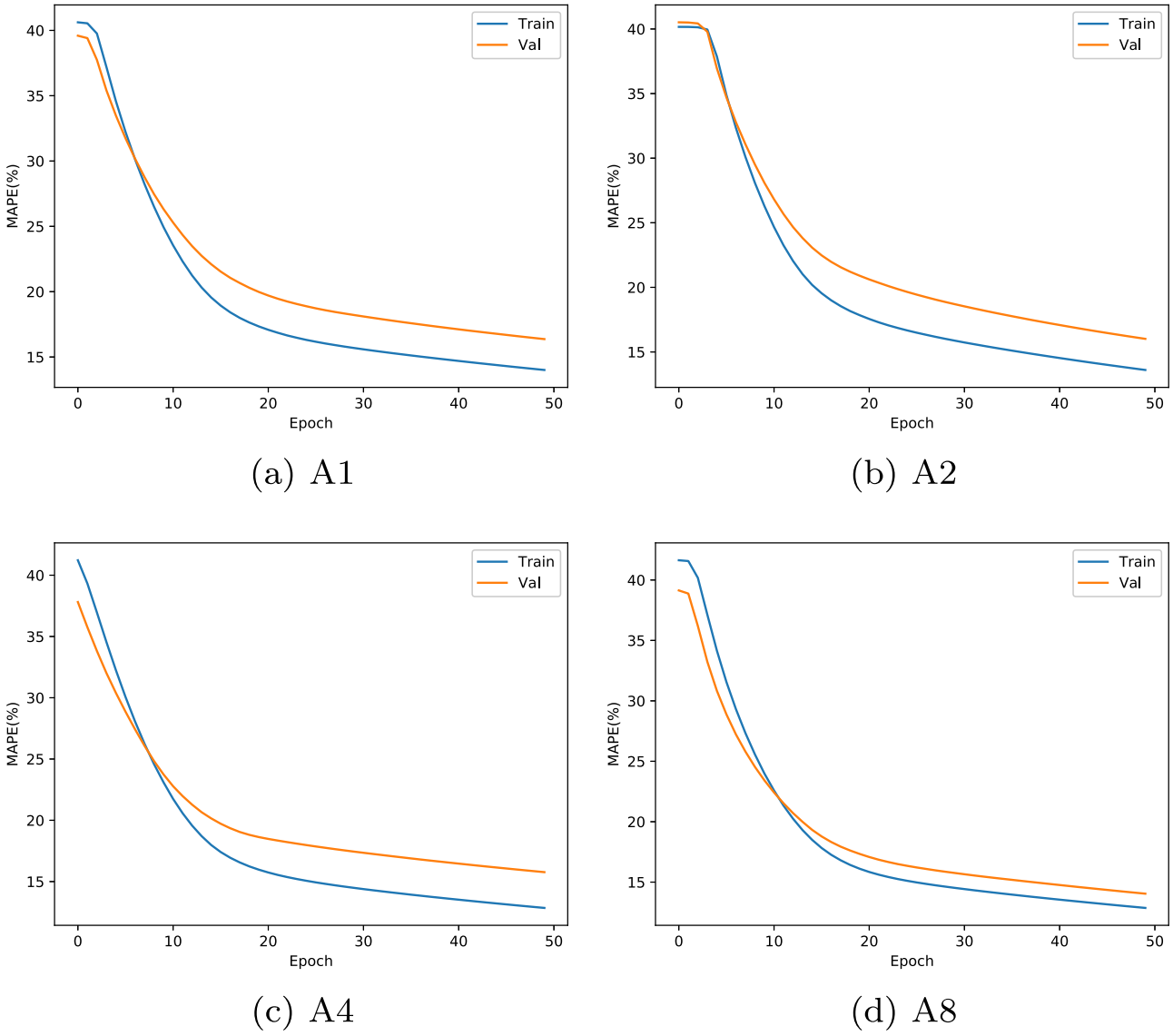

FIG. 6. The MAPE on the training set and the validation set for the training process.

the traffic state changes dramatically, which largely decreases the forecasting performance. The nonparametric methods, such as the ANN, optimize the parameters during backpropagation. However, the mean square error magnifies the effect of the non-Gaussian noises, such as outliers. In this regard, this kind of methods could hardly achieve superior performance for real-world cases, since the non-Gaussian noises are inevitable under this situation. The NiLSTM addresses this issue by equipping a correntropy metric loss, which empowers the NiLSTM immune to the non-Gaussian noises. The deep learning model, e.g., SAE, performs best among the control models except for the NiLSTM. The NiLSTM achieves the lowest RMSE than other models on all the datasets, and almost outperforms other models in MAPE, except the A1 dataset. The MAPE of the NiLSTM is only 0.08% higher than that of the SAE.

3. Evaluation of NiLSTM with different kernel sizes

σ is a hyperparameter for the correntropy. We set the length of the input traffic flow sequence to 12 as the above experiments to evaluate the kernel size that affects the performance of the NiLSTM for traffic flow forecasting.

We set the range of kernel size to $\{0.1, 0.2, 0.5, 1.0, 2.0, 3.0\}$, which follows the suggestion in Liu *et al.*[39] Table IV exhibits the performance of the NiLSTM with different kernel sizes on the four

datasets. The experimental results demonstrate the robustness of the NiLSTM model with the suggested kernel sizes. Although there are slight differences in the forecasting performance with different kernel sizes, this should not obscure the fact that the forecasting performance of the LSTM model has been improved equipped with the noise-immune loss.

We also draw the learning curve of the NiLSTM on the training set and validation set. The length of the input sequence is set to 12 in this experiment. Figures 5 and 6 show the RMSE and the MAPE, respectively. These two figures both contain four sub-figures, which correspond to the four datasets, e.g., A1, A2, A4, and A8. Although there are subtle differences, the errors on the training set and the validation set decrease fast for the first 15 epochs. After that, the decreases in the errors slow down and gradually tend to become stable. Meanwhile, the NiLSTM performs slightly better on the training set than on the validation set, which accords to the common sense. Overall, our model performs normally without overfitting.

Our noise-immune loss is general and can be integrated in other network architectures and other applications, such as biomedical computing,[51–53] intelligent computing,[54,55] and algebraic immunity.[56–59]

IV. CONCLUSIONS

In this paper, we propose a noise-immune LSTM network for short-term traffic flow forecasting. The conventional LSTM networks are sensitive to non-Gaussian noises mixed in the traffic flow sequences, which affects the forecasting performance. The proposed NiLSTM eliminates the effectiveness of non-Gaussian noises inside the traffic flow by employing the correntropy criterion as the loss function. Extensive experiments are designed to illustrate the effectiveness of the NiLSTM.

ACKNOWLEDGMENTS

This work is supported by the National Science Foundation of China (NSFC) (Grant No. 61902232), the Natural Science Foundation of Guangdong Province (Nos. 2018A030313291 and 2018A030313889), the Education Science Planning Project of Guangdong Province (No. 2018GXJK048), the STU Scientific Research Foundation for Talents (No. NTF18006), the Guangdong Special Cultivation Funds for College Students' Scientific and Technological Innovation (No. pdjh2020b0222), and the grant from the Hong Kong Polytechnic University (No. 1ZE8J).

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

DATA AVAILABILITY

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

REFERENCES

[1]Y. Zhang, S. Wang, B. Chen, J. Cao, and Z. Huang, "Trafficgan: Network-scale deep traffic prediction with generative adversarial nets," IEEE Trans. Intell. Transp. Syst. 1–12 (2019).
[2]A. Fragkou, T. Karakasidis, and E. Nathanail, "Detection of traffic incidents using nonlinear time series analysis," Chaos **28**, 063108 (2018).
[3]J. Villalobos, V. Muñoz, J. Rogan, R. Zarama, J. F. Penagos, B. Toledo, and J. A. Valdivia, "Modeling a bus through a sequence of traffic lights," Chaos **25**, 073117 (2015).
[4]Z. Zhang, Y. Sheng, Z. Hu, and G. Chen, "Optimal and suboptimal networks for efficient navigation measured by mean-first passage time of random walks," Chaos **22**, 043129 (2012).
[5]E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Short-term traffic forecasting: Where we are and where we're going," Transp. Res. Part C Emerg. Technol. **43**, 3–19 (2014).
[6]Y. Lv, Y. Chen, X. Zhang, Y. Duan, and N. L. Li, "Social media based transportation research: The state of the work and the networking," IEEE/CAA J. Autom. Sin. **4**, 19–26 (2017).
[7]L. Li, Y. Lv, and F. Wang, "Traffic signal timing via deep reinforcement learning," IEEE/CAA J. Autom. Sin. **3**, 247–254 (2016).
[8]U. Mori, A. Mendiburu, M. Álvarez, and J. A. Lozano, "A review of travel time estimation and forecasting for advanced traveller information systems," Transportmetrica A Transport Sci. **11**(2), 119–157 (2015).
[9]I. Kaysi, M. Ben-Akiva, and H. Koutsopoulos, "Integrated approach to vehicle routing and congestion prediction for real-time driver guidance," Transp. Res. Rec. **1408**, 66–74 (1993).
[10]K. Y. Chan, T. S. Dillon, J. Singh, and E. Chang, "Neural-network-based models for short-term traffic flow forecasting using a hybrid exponential smoothing and Levenberg–Marquardt algorithm," IEEE Trans. Intell. Transp. Syst. **13**(2), 644–654 (2012).
[11]D. Tikunov and T. Nishimura, "Traffic prediction for mobile network using Holt-Winter's exponential smoothing," in *2007 15th International Conference on Software, Telecommunications and Computer Networks* (IEEE, 2007), pp. 1–5.
[12]T. Zhou, D. Jiang, Z. Lin, G. Han, X. Xu, and J. Qin, "Hybrid dual Kalman filtering model for short-term traffic flow forecasting," IET Intell. Transp. Syst. **13**, 1023–1032 (2019).
[13]L. Cai, Z. Zhang, J. Yang, Y. Yu, T. Zhou, and J. Qin, "A noise-immune Kalman filter for short-term traffic flow forecasting," Phys. A Stat. Mech. Appl. **536**, 1–9 (2019).
[14]S. Zhang, Y. Song, D. Jiang, T. Zhou, and J. Qin, "Noise-identified Kalman filter for short-term traffic flow forecasting," in *The 15th International Conference on Mobile Ad-Hoc and Sensor Networks* (IEEE, 2019), pp. 1–5.
[15]H. Zare Moayedi and M. A. Masnadi-Shirazi, "Arima model for network traffic prediction and anomaly detection," in *2008 International Symposium on Information Technology* (IEEE, 2008), Vol. 4, pp. 1–6.
[16]L. J. Yu Peng, M. Lei, and P. XiYuan, "A novel hybridization of echo state networks and multiplicative seasonal arima model for mobile communication traffic series forecasting," Neural Comput. Appl. **24**, 883–890 (2014).
[17]B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results," J. Transp. Eng. **129**(6), 664–672 (2003).
[18]Y. Zhang, Y. Zhang, and A. Haghani, "A hybrid short-term traffic flow forecasting method based on spectral analysis and statistical volatility model," Transp. Res. Part C Emerg. Technol. **43**, 65–78 (2014).
[19]B. Ghosh, B. Basu, and M. O'Mahony, "Multivariate short-term traffic flow forecasting using time-series analysis," IEEE Trans. Intell. Transp. Syst. **10**(2), 246–254 (2009).
[20]H. Liu, Z. Canfang, L. Jiansha, L. Mian, Z. Shusheng, J. Yuyang, and Z. Yufen, "Simultaneous measurement of trace monoadenosine and diadenosine monophosphate in biomimicking prebiotic synthesis using high-performance liquid chromatography with ultraviolet detection and electrospray ionization mass spectrometry characterization," Anal. Chim. Acta **566**(1), 99–108 (2006).
[21]J. Z. Zhu, J. X. Cao, and Y. Zhu, "Traffic volume forecasting based on radial basis function neural network with the consideration of traffic flows at the adjacent intersections," Transp. Res. Part C Emerg. Technol. **47**, 139–154 (2014).
[22]W. Cai, J. Yang, Y. Yu, Y. Song, T. Zhou, and J. Qin, "Pso-elm: A hybrid learning model for short-term traffic flow forecasting," IEEE Access **8**, 6505–6514 (2020).
[23]L. Cai, Y. Yu, S. Zhang, Y. Song, Z. Xiong, and T. Zhou, "A sample-rebalanced outlier-rejected k-nearest neighbour regression model for short-term traffic flow forecasting," IEEE Access **8**, 22686–22696 (2020).
[24]W.-C. Hong, Y. Dong, F. Zheng, and C.-Y. Lai, "Forecasting urban traffic flow by SVR with continuous ACO," Appl. Math. Model. **35**(3), 1282–1291 (2011).

[25]W. Cai, D. Yu, Z. Wu, X. Du, and T. Zhou, "A hybrid ensemble learning framework for basketball outcomes prediction," Phys. A Stat. Mech. Appl. **528**, 121461 (2019).
[26]L. Cai, Q. Chen, W. Cai, X. Xu, T. Zhou, and J. Qin, "Svrgsa: A hybrid learning based model for short-term traffic flow forecasting," IET Intell. Transp. Syst. **13**(9), 1348–1355 (2019).
[27]Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature **521**, 436 (2015).
[28]W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: Deep belief networks with multitask learning," IEEE Trans. Intell. Transp. Syst. **15**(5), 2191–2201 (2014).
[29]T. Zhou, G. Han, X. Xu, Z. Lin, C. Han, Y. Huang, and J. Qin, "δ-agree AdaBoost stacked autoencoder for short-term traffic flow forecasting," Neurocomputing **247**, 31–38 (2017).
[30]T. Zhou, G. Han, X. Xu, C. Han, Y. Huang, and J. Qin, "A learning-based multimodel integrated framework for dynamic traffic flow forecasting," Neural Process. Lett. **49**, 407–430 (2019).
[31]Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Wang, "Traffic flow prediction with big data: A deep learning approach," IEEE Trans. Intell. Transp. Syst. **16**(2), 865–873 (2015).
[32]P. Lingras, S. Sharma, and M. Zhong, "Prediction of recreational travel using genetically designed regression and time-delay neural network models," Transp. Res. Rec. **1805**(2), 16–24 (2002).
[33]R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13 (JMLR.org, 2013), pp. III-1310–III-1318.
[34]S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Comput. **9**, 1735–1780 (1997).
[35]F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," in *1999 Ninth International Conference on Artificial Neural Networks ICANN 99 (Conference Publication No. 470)* (IET, 1999), Vol. 2, pp. 850–855.
[36]X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," Transp. Res. Part C Emerg. Technol. **54**, 187–197 (2015).
[37]T. Yongxue and P. Li, "Predicting short-term traffic flow by long short-term memory recurrent neural network," in *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)* (IEEE, 2015), pp. 153–158.
[38]L. Canyi, J. Tang, M. Lin, L. Lin, S. Yan, and Z. Lin, "Correntropy induced l2 graph for robust subspace clustering," in *2013 IEEE International Conference on Computer Vision* (IEEE, 2013), pp. 1801–1808.
[39]W. Liu, P. P. Pokharel, and J. C. Príncipe, "Correntropy: Properties and applications in non-Gaussian signal processing," IEEE Trans. Signal Process. **55**, 5286–5298 (2007).
[40]R. He, W. Zheng, and B. Hu, "Maximum correntropy criterion for robust face recognition," IEEE Trans. Pattern Anal. Mach. Intell. **33**(8), 1561–1576 (2011).
[41]R. J. Bessa, V. Miranda, and J. Gama, "Entropy and correntropy against minimum square error in offline and online three-day ahead wind power forecasting," IEEE Trans. Power Syst. **24**(4), 1657–1666 (2009).
[42]R. He, B. Hu, W. Zheng, and X. Kong, "Robust principal component analysis based on maximum correntropy criterion," IEEE Trans. Image Process. **20**(6), 1485–1494 (2011).
[43]A. Garde, L. Sörnmo, R. Jané, and B. F. Giraldo, "Correntropy-based spectral characterization of respiratory patterns in patients with chronic heart failure," IEEE Trans. Biomed. Eng. **57**(8), 1964–1972 (2010).
[44]B. Chen, L. Xing, H. Zhao, N. Zheng, and J. C. Príncipe, "Generalized correntropy for robust adaptive filtering," IEEE Trans. Signal Process. **64**, 3376–3387 (2016).
[45]Y. Wang, J. H. van Schuppen, and J. Vrancken, "Prediction of traffic flow at the boundary of a motorway network," IEEE Trans. Intell. Transp. Syst. **15**(1), 214–227 (2014).
[46]B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)* (IJCAI, 2018).
[47]S. Arlot and A. Celisse, "A survey of cross-validation procedures for model selection," Stat. Surv. **4**, 40–79 (2010).
[48]F. Chollet *et al.*, see https://keras.io for "Keras" (2015).
[49]M. Lippi, M. Bertini, and P. Frasconi, "Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning," IEEE Trans. Intell. Transp. Syst. **14**(2), 871–882 (2013).
[50]Y. Xie, Y. Zhang, and Z. Ye, "Short-term traffic volume forecasting using Kalman filter with discrete wavelet decomposition," Comp. Aided Civil Infrastruct. Eng. **22**(5), 326–334 (2007).
[51]T. Zhou, G. Han, B. N. Li, Z. Lin, E. J. Ciaccio, P. H. Green, and J. Qin, "Quantitative analysis of patients with celiac disease by video capsule endoscopy: A deep learning method," Comput. Biol. Med. **85**, 1–6 (2017).
[52]B. N. N. Li, X. Wang, R. Wang, T. Zhou, R. Gao, E. J. Ciaccio, and P. H. Green, "Celiac disease detection from videocapsule endoscopy images using strip principal component analysis," IEEE/ACM Trans. Comput. Biol. Bioinform. 1–10 (2019).
[53]Y. Song, Z. Yu, T. Zhou, J. Y.-C. Teoh, B. Lei, C. Kup-Sze, and J. Qin, "CNN in CT image segmentation: Beyond loss function for exploiting ground truth images," in *2020 IEEE International Symposium on Biomedical Imaging (ISBI)* (IEEE, 2020), pp. 1–4.
[54]D. Jiang, K. Wu, D. Chen, G. Tu, T. Zhou, A. Garg, and L. Gao, "A probability and integrated learning based classification algorithm for high-level human emotion recognition problems," Measurement **150**, 107049 (2019).
[55]D. Jiang, Z. Liu, L. Zheng, and J. Chen, "Factorization meets neural networks: A scalable and efficient recommender for solving the new user problem," IEEE Access **8**, 18350–18361 (2020).
[56]Y. Chen, F. Guo, Z. Gong, and W. Cai, "One note about the Tu-Deng conjecture in case w(t)=5," IEEE Access **7**, 13799–13802 (2019).
[57]Y. Chen, L. Zhang, D. Tang, and W. Cai, "Translation equivalence of Boolean functions expressed by primitive element," IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **102**, 672–675 (2019).
[58]Y. Chen, F. Guo, and J. Ruan, "Constructing odd-variable RSBFS with optimal algebraic immunity, good nonlinearity and good behavior against fast algebraic attacks," Discrete Appl. Math. **262**, 1–12 (2019).
[59]Y. Chen, F. Guo, H. Xiang, W. Cai, and X. He, "Balanced odd-variable RSBFS with optimum AI, high nonlinearity and good behavior against FAAS," IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **102**, 818–824 (2019).