Wide-Attention and Deep-Composite Model for Traffic Flow Prediction in Transportation Cyber-Physical Systems

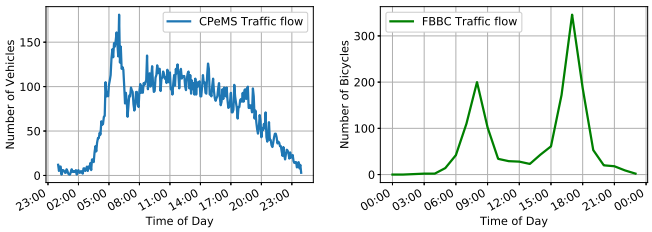
Junhao Zhou, Hong-Ning Dai, *Senior Member, IEEE*, Hao Wang, *Member, IEEE* and Tian Wang

***Abstract*—Recently, traffic flow prediction has drawn significant attention because it is a prerequisite in intelligent transportation management in urban informatics. The massively-available traffic data collected from various sensors in Transportation Cyber-Physical Systems brings the opportunities in accurately forecasting traffic trend. Recent advances in deep learning shows the effectiveness on traffic flow prediction though most of them only demonstrate the superior performance on traffic data from a single type of vehicular carriers (e.g., cars) and does not perform well in other types of vehicles. To fill this gap, we propose a wide-attention and deep-composite (WADC) model consisting of a wide-attention module and a deep-composite module in this paper. In particular, the wide-attention module can extract global key features from traffic flows via a linear model with self-attention mechanism. The deep-composite module can generalize local key features via Convolutional Neural Network component and Long Short-Term Memory Network component. We also perform extensive experiments on different types of traffic flow datasets to investigate the performance of WADC model. Our experimental results exhibit that WADC model outperforms other existing approaches.**

I. Introduction

WE have recently witnessed the rapid advances in transportation cyber-physical systems (TCPS) with provision of convenient and efficient traffic management. There are also diverse travelling manners from vehicular driving to bicycle-riding. Meanwhile, the increased traffic flows also result in the traffic congestion problem, which has become one of the main obstacles for urban development [1], [2]. The deployment of TCPS brings the opportunities to address this problem. In particular, the proliferation of traffic data collected from traffic sensors, instruments and other transportation facilities in TCPS also brings opportunities in overcoming the traffic congestion, making prediction and enforcing precaution in advance after analyzing the massive traffic data.

The core of traffic prediction is to predict the traffic tendency for the next time interval via analyzing the historical traffic data. In the past decades, a number of studies started to adopt machine learning (ML) approaches to investigate the extensive traffic data for traffic flow prediction. For example, support vector regression (SVR) is one of the traditional ML methods to predict the short-term traffic flow [3]. Meanwhile, Artificial Neural Network (ANN) [4] is employed in predicting traffic flow time series. However, the traditional ML algorithms cannot capture the complex non-linear spatial-temporal dependency from traffic flow data.



(a) Vehicular traffic flow (CPeMS) (b) Cycling traffic flow (FBBC)

Fig. 1. Traffic flow comparison between vehicular data and cycling data

Recently, deep learning (DL) models have shown the advantages in extracting valuable information from massive traffic data and outperforming other ML models in terms of accuracy. In particular, Convolutional Neural Network (CNN) model demonstrates the outstanding performance in predicting the vehicle speed after converting vehicular traffic to images, which are then processed by CNN models [5]. Meanwhile, Long Short-Term Memory Network (LSTM) shows the advantages in forecasting traffic speed via learning the temporal features from traffic flow data [6].

However, previous ML and DL approaches used in traffic prediction have their limitations. In particular, most of ML and DL approaches only adopt a sole ML or DL model so that they may perform excellent on a specific dataset while performing worse on another traffic dataset. Our urban transportation systems essentially consist of different types of transportation carriers. The traffic peaks at different transportation carriers may be different from each other. Take Fig. 1 as an example, in which there are two kinds of traffic flows: vehicular traffic as shown in Fig. 1(a) and cycling traffic as shown in Fig. 1(b), where the traffic flow is represented by the number of vehicles or bicycles during the t-th time interval from source A to destination B. Note that both vehicular traffic and cycling

The work described in this paper was partially supported by Macao Science and Technology Development Fund under Macao Funding Scheme for Key R & D Projects (0025/2019/AKP).

J. Zhou and H.-N. Dai are with Macau University of Science and Technology, Macau SAR (email: junhao_zhou@qq.com; hndai@ieee.org).

H. Wang is Department of Computer Science, Norwegian University of Science and Technology, Gjøvik, Norway (email: hawa@ntnu.no).

T. Wang is College of Computer Science and Technology, Huaqiao University, Xiamen, China (email: cs_tianwang@163.com).

traffic are obtained from realistic datasets[1]. We can observe from Fig. 1 that cycling flow data and vehicular flow data have different peak hours while cycling flow data has more significant fluctuations than vehicular flow data. For example, the peak hour for vehicular flow is about 7:30 am in the morning while the peak hour for cycling flow is 5:30 pm in the afternoon. Moreover, vehicular traffic may last a longer period of peak hours than cycling traffic. The reason may lie in the fact that cycling is more like a sport being more susceptible to weather conditions than driving vehicles, which is less influenced by weather conditions. In addition, vehicular traffic is often affected by many conditions, such as transportation facility settings (the number of lanes, traffic lights, the volume of carriers) and holidays (or special events). In contrast, cycling traffic is less influenced by the above factors since bicycles have often been ridden on special bike lanes, where there are no traffic lights.

It is necessary to design a DL model to better analyze traffic flow data with consideration of various types of traffic flows. To this end, we originally propose a wide-attention and deep-composite (WADC) model. The main research contributions of the paper can be summarized as follows.

- We put forth a composite DL model to analyze different types of traffic flows. In particular, our WADC model consists of a wide-attention module and a deep-composite module. The wide-attention module can extract global key features from traffic flow data. Different from wide module used in existing studies, we leverage the attention mechanism based on L_1 and L_2 regularizations so as to better capture different features from different types of traffic flows. Meanwhile, the deep-composite module consisting of LSTM and CNN components is beneficial to learn complicated features while requiring less feature engineering.
- We run a number of experiments to investigate the performance of our WADC model. Specially, we adopt different types of realistic traffic flow datasets (i.e., vehicular traffic flow and cycling traffic flow). Meanwhile, we investigate the performance of our WADC model with comparison with other nine representative baseline approaches. The experimental results exhibit that our proposed model outperforms than existing ML and DL models. In addition, we further investigate the impacts of various parameters, such as regularization methods, the varied number of convolutional filters and LSTM neurons, in WADC model. Consequently, experimental results on different type traffic flow datasets have demonstrated that our model has the advantage in *generalization*, i.e., the ability of being adapted to different type traffic flows.

The rest of the paper is organized as follows. Section II presents the studies on tradition ML and DL models. Section III describes the main proposed methods in details. Section IV gives the experimental evaluation results. In Section V, we conclude this paper and outline future research directions.

[1]Caltrans Performance Measurement System (CPeMS) from California, USA, http://pems.doc.ca.gov; Fremont Bridge Bicycle Counts (FBBC) at Seattle, USA https://data.seattle.gov.

II. Related work

This section reviews recent advances in traffic flow prediction. We classify recent research into two types: traditional machine learning approaches (in Sec. II-A) and deep learning approaches (in Sec. II-B).

A. Traditional machine learning

Traditional traffic flow prediction models include parametric models such as Autoregressive Integrated Moving Average model (ARIMA) [7], Kalman filtering model [8] and their extensions. For example, EMD-ARIMA model [9], Seasonal ARIMA [10], an integration of Kalman filter and ARIMA [11] have been reported for traffic flow prediction in literature. These time series prediction methods have been developed to assist Transportation Cyber-Physical Systems (TCPS) to analyze traffic flow and predict the traffic trend.

Besides ARIMA, Kalman filtering model and their variants, non-parametric machine learning (ML) approaches have been widely used in traffic prediction in TCPS. The representative ML methods include Support vector regression (SVR), deep regression (DR) and Artificial Neural Network (ANN) that have been adopted in the traffic prediction through learning from massive traffic data. In particular, SVR was proposed for short-term traffic flow forecasting in [3], which is an online learning weighted algorithm. Meanwhile, the work of [12] shows that ANN has been applied in short-term traffic flow forecasting and shown a superior performance than other traditional ML approaches.

B. Deep learning approaches

Compared with traditional ML, deep learning (DL) approaches provide a promising way in capturing the complex features from a huge volume of data so as to have diverse industrial applications, such as sentimental analysis [13] and electricity-theft detection [14]. Furthermore, DL approaches have also been applied in traffic prediction. For example, ref. [15] proposed a deep neural network architecture model using auto-encoders as building blocks to learn the features for traffic flow prediction. As in [16], the authors proposed a traffic forecast model on top of LSTM network. The work of [17] applied gated recurrent neural network (GRU) to predict urban traffic flow with consideration of weather conditions.

Over the last several years, composite DL methods have drawn significant attention since composite DL methods have the advantages in extracting various features from traffic flow data [18]. In particular, ref. [19] combined CNN and LSTM to construct a LSTM-CNN model to extract the spatial-temporal features from the traffic flow data. Ref. [20] proposed a Wide & Deep (namely W&D) framework for jointly training deep neural networks. As in [21], the authors present a Deep & Cross Network (DCN) to improve the efficiency in learning all types of features. Moreover, the composite DL models have shown their advantages in traffic flow forecasting. To address the challenges of spatial and temporal dependency, a

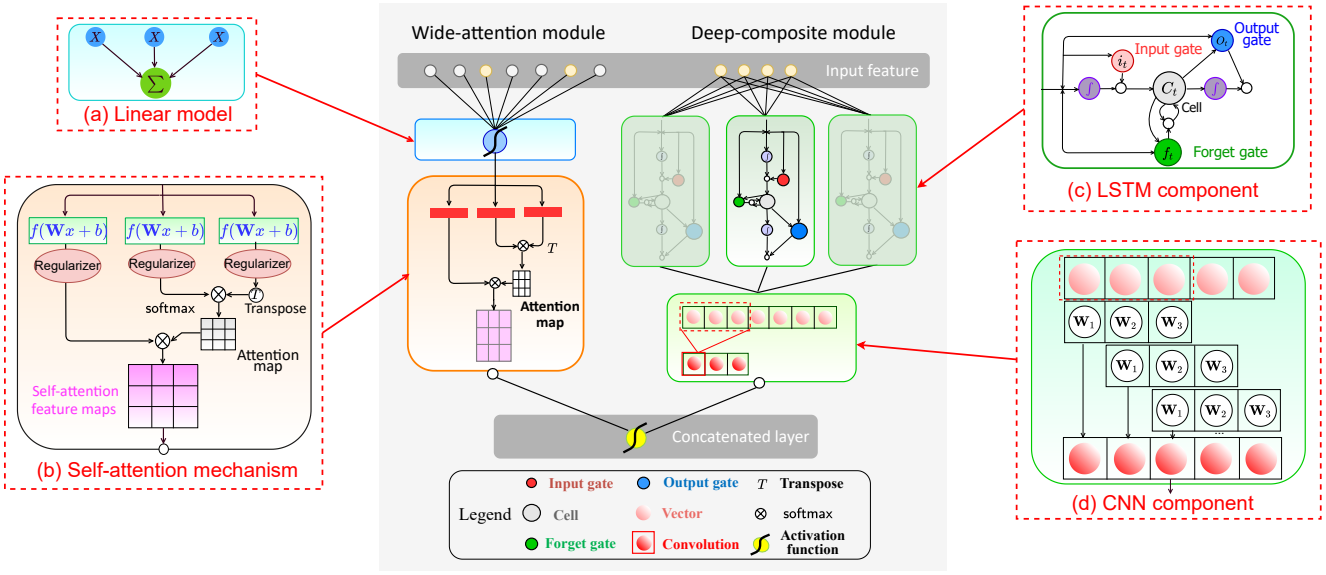


Fig. 2. Wide-attention and deep-composite (WADC) model

novel model called Diffusion Convolutional Recurrent Neural Network (DCRNN) was proposed by Li et al. [22]. Ref. [23] designs an end-to-end learning architecture called Fusion Convolutional Long Short-term Memory Network (FCL-Net) which merges convolution processing and LSTM structure, to address the passenger demand problem. Furthermore, ref. [24] shows that the composite models present the excellent performance than the sole DL models.

Therefore, we summarize the existing approaches having two major limitations. First, most of traditional ML methods are suffering from efforts in data preprocessing and low prediction accuracy. Second, most of these DL models can only capture partial features while missing multi-dimensional data. Motivated by recent improvement in composite DL methods, we introduce a WADC composite structure to predict the different types of traffic flow data. Moreover, compared with tradition ML and DL models, WADC can concentrate on key global features via wide-attention module and extract key local features from deep-composite module, thereby achieving significant performance improvement.

III. Our Approach

In this paper, we propose a wide-attention and deep-composite (namely WADC) model as shown in Fig. 2. Sections III-A and III-B then give the details of wide-attention module and deep-composite module, respectively.

A. Wide-attention module

As shown in Fig. 2, WADC model consists of wide-attention module and deep-composite module. In particular, as shown in Figs. 2(a) and (b) (i.e., magnified views), the wide-attention module contains two major components: a linear model and a self-attention mechanism. In the linear model, x_i denotes a specific traffic flow and $f(x_i)$ is the prediction function of traffic flow x_i. The relationship between $f(x_i)$ and x_i can be expressed as the following equation,

$$f(x_i) = \mathbf{W}x_i + b, \tag{1}$$

where $\mathbf{W}$ denotes the weight value and b denotes the bias value. The linear module is beneficial to capture the relationships between individual features via using simple feature engineering. Specifically, feature engineering can generate various derived features from the traffic flow. However, the embedding matrix f suffers from the redundancy if the linear model always obtains the approximate sum of weights each time.

Therefore, we adopt regularization optimization methods to normalize the weights and mitigate the bias. The regularization process can enhance the diversity of the sum values of weight vectors across different individual features. The regularization methods mainly include kernel-regularizer and bias-regularizer in the neural network. The kernel-regularizer punishes the weights and the bias-regularizer reduces the bias. In particular, we mainly leverage these regularizers to improve the diversity of attention mechanism and avoid overfitting for traffic flow data regression. We first use kernel-regularizer to regularize the weight matrices via L_2 regularization [25]. The main idea of L_2 regularization is basically to minimize the sum of the squared differentiation (denoted by S) between the target weight and the estimated weight denoted by $\mathbf{W}$ and $\mathbf{W}_i$, respectively. The L_2 regularization can expressed as follows,

$$S = \sum_{i=1}^{N} (\mathbf{W} - \mathbf{W}_i)^2. \tag{2}$$

We then employ bias-regularizer to regularize the bias value via L_1 regularization. The main idea of L_1 regularization is to minimize the sum of the absolute differences (denoted by S') between the targeted bias and estimated bias denoted by b and b_i, respectively. The L_1 regularization can expressed as

follows,

$$S' = \sum_{i=1}^{N} |b - b_i|. \tag{3}$$

The second key component in the wide-attention module is self-attention mechanism. In particular, self-attention mechanism shows a superior equilibrium between the capability of modeling long-term dependencies and the efficiency of computation and statistics. For example, we utilize self-attention mechanism in the feature map to calculate the feature response representing a weighted sum of the features at all positions. Therefore, we can calculate the attention weight with a low computational cost. This mathematical expression of self-attention mechanism is shown as follows,

$$e_{ij} = \frac{f^T(x_i) f(x_j)}{\sqrt{d_k}}, \tag{4}$$

where e_{ij} denotes the relation between i-th value and j-th value and the transposed matrix of $f(x_i)$ is denoted by $f^T(x_i)$. As shown in Eq. (4), the score of *attention map* is divided by $\sqrt{d_k}$ representing the square root of vector dimension of matrix $f(x_i)$. Moreover, compared with previous methods, this step can lead to a faster convergence.

In addition, the attention weight of the i-th value corresponding to j-th value is also denoted by a_{ij}, which is given as follows,

$$a_{ij} = \mathsf{softmax}(e_{ij}) = \frac{\exp(e_{ij})}{\sum_j \exp(e_{ij})}. \tag{5}$$

Moreover, we denote *regularization self-attention feature maps* representing the output of wide-attention module by $\mathbf{Y}_{\text{wide}}$, which is given by the following equation,

$$\mathbf{Y}_{\text{wide}} = a_{ij} f(x_i) = \mathsf{softmax}(e_{ij}) f(x_i). \tag{6}$$

B. Deep-composite module

The deep-composite module consists of an LSTM component and a CNN component, as shown in Figs. 2(c) and (d), respectively. The LSTM component can essentially learn the temporal sequential dependency and the CNN component enables to extract the features from one-dimensional (1D) sequences traffic flow data. We then discuss the details of these components as follows.

1) LSTM component: LSTM network is a variant of recurrent neural network (RNN) to settle the exploding and vanishing gradient problem [26]. Basically, a common LSTM unit is composed of a **cell** c_t and three gates (i.e., **input/output gates**, a **forget gate**), controlling the new information stored by the input gate and the previous information discarded by the forget gate. Thanks to the learning capability of long-term dependencies, LSTM has the strength of capturing the temporal sequential dependency in traffic flow prediction. The output values of LSTM are calculated as follows,

$$i_t = \sigma(\mathbf{W}_i x_t + \mathbf{U}_i h_{t-1} + b_i), \tag{7}$$

$$f_t = \sigma(\mathbf{W}_f x_t + \mathbf{U}_f h_{t-1} + b_f), \tag{8}$$

$$o_t = \sigma(\mathbf{W}_o x_t + \mathbf{U}_o h_{t-1} + b_o), \tag{9}$$

$$\theta_t = \tanh(\mathbf{W}_s x_t + \mathbf{U}_s h_{t-1} + b_s), \tag{10}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \theta_t, \tag{11}$$

$$h_t = o_t \odot \sigma(c_t), \tag{12}$$

where i_t, f_t and o_t denote input gate, forget gate and output gate, respectively. Meanwhile, θ_t is a $\tanh$ layer to create a vector consisting of new candidate value in the cell state. We also denote Hadamard product by operator $\odot$. The hidden state vector and the cell state vector are denoted by h_{t-1} and c_t, respectively. In particular, $\mathbf{W}_p$, $\mathbf{U}_p$, and b_p ($p \in \{i, f, o, s\}$) are the learning parameters.

2) CNN component: Compared with convention artificial neural network (ANN), the advantage of CNN is that each neuron connects to its bordering neurons rather than all neurons. Meanwhile, CNN can extract and map the internal features from one-dimensional sequences data (denoted by CNN-1D). Due to the excellent ability of identifying simple time sequences, we then employ the CNN-1D layer to capture key features from time sequences of traffic flow data. Moreover, the CNN-1D layer only brings a low computational complexity computational cost due the simple 1-D CNN structure.

The CNN-1D layer takes $\mathbf{Y}_k^l$ as input $\mathbf{Y}_k^0$, where $\mathbf{Y}_k^0 = h_t$. The formulation of each convolutional layer l can be computed as follows,

$$\mathbf{Y}_{\text{deep}} = \mathbf{Y}_k^l = f(\sum_{i=1}^{l-1} \mathsf{conv1D}(\mathbf{W}_k^l, \mathbf{Y}_k^{l-1}) + b_k^l), \tag{13}$$

where $\mathbf{Y}_{\text{deep}}$ is the output of deep-composite module, $\mathbf{W}_k^l$ and b_k^l denote the weight and bias at the k^{th} neuron at layer l, respectively. In particular, $\mathsf{conv1D}(\cdot)$ is used to perform one-dimensional convolution and $f(\cdot)$ is an activation function.

C. Concatenated layer

Finally, the concatenated layer combines the outputs of the wide-attention module and the deep-composite module. Then, it exports the final traffic flow prediction via feeding the combined features to the activation function. In particular, the concatenated layer can deal with the weighted sum of the out of these two modules together and optimize the learning parameters, simultaneously. The concatenated layer mainly consists of a fully-connected layer. We denote the predicted traffic flow by $\mathbf{Y}$, which can be calculated as follows,

$$\mathbf{Y} = f(\mathbf{W}_c \cdot \mathsf{Concat}[\mathbf{Y}_{\text{wide}}, \mathbf{Y}_{\text{deep}}] + b_c), \tag{14}$$

where $\mathsf{Concat}[\cdot]$ denotes a concatenated function to combine $\mathbf{Y}_{\text{wide}}$ and $\mathbf{Y}_{\text{deep}}$. In the concatenated layer, the learning parameters for weight and bias are denoted by $\mathbf{W}_c$ and b_c, respectively.

D. Algorithm analysis

Algorithm 1 summarizes the whole working procedure of our proposed WADC model. In particular, our model can extract different features via wide-attention module (lines 1-5) and generalize the deep feature combinations via deep-composite module (lines 6-10). Consequently, the robustness of model can be improved and the computational cost for traffic flow prediction can be reduced. We then combine results $\mathbf{Y}_{\text{wide}}$

Algorithm 1 WADC model training procedure

Input: Input features: x_i and x_t
Output: Traffic flow prediction
1: **function** WIDE-ATTENTION MODULE(x_i)
2: *Linear model:* $f(x_i) = \mathbf{W}(x_i) + b$,
3: *Self-attention:* Balance long-range dependencies and computational.
4: **return** wide features: $\mathbf{Y}_{\text{wide}}$
5: **end function**
6: **function** DEEP-COMPOSITE MODULE(x_t)
7: *LSTM component:* Capture the temporal sequential feature.
8: *CNN component:* Extract the features from 1D sequences.
9: **return** deep features: $\mathbf{Y}_{\text{deep}}$
10: **end function**
11: **function** CONCATENATED LAYER($\mathbf{Y}_{\text{wide}}$, $\mathbf{Y}_{\text{deep}}$)
12: Concatenate wide and deep features to fully-connected layer
13: $\mathbf{Y} = f(\mathbf{W}_c \cdot \mathsf{Concat}[\mathbf{Y}_{\text{wide}}, \mathbf{Y}_{\text{deep}}] + b_c)$
14: **end function**
15: Optimizer: RMSProp

and $\mathbf{Y}_{\text{deep}}$ via the concatenated layer (lines 11-14). Finally, We select the root mean square prop (RMSProp) optimizer to minimize the square errors between the prediction value and the actual target value (line 15).

IV. EXPERIMENT

A. Experimental Settings

We perform the experiments on an Intel Core i7-7700HQ CPU and 16 GB Memory (RAM). In particular, we employ a Nvidia GTX 1050 GPU with 4 GB GPU Memory to improve the effect of training phase. In the framework, we use Keras 2.0 (i.e., Tensorflow as backend) with Python 2.6 in Ubuntu 16.04 platform.

1) Dataset description: We run the experiments on two widely-utilized datasets in traffic flow prediction. One is CPeMS dataset. The CPeMS dataset is collected every 30 seconds in real-time from more than 15,000 detectors, which have been deployed in highway systems across major urban areas of California, USA. We excerpt a subset of data records from a certain highway in CPeMS, in which there are 12,096 road driving records and the length of each time interval is chosen as 5 minutes (from Jan. 04, 2016 to Mar. 31, 2016).

Another dataset is Fremont Bridge Bicycle Counter (FBBC) dataset. It records the number of bikes, crossing the Fremont bridge (at Portland, Oregon) on the pedestrian or bicycle pathways. In particular, the detectors on the pathways count the passing of bicycles regardless of travel direction. The FBBC dataset contains 59,832 records and each time interval is chosen as 1 hour (from Oct. 03, 2012 to July. 31, 2019).

2) Training settings: In our experiment, each dataset is divided into two subsets: 1) the training set and 2) the validation set. For example, the training set of CPeMS contains 7,776 traffic flow records from Jan. 04, 2016 to Feb. 29, 2016 and the validation set of CPeMS contains the remaining 4,320 records. Another training set of FBBC contains 54,744 records from Oct. 03, 2012 to Dec. 31, 2018 and the validation set contains the remaining 5,088 records.

3) Performance metrics: In our traffic flow prediction experiments, we adopt three metrics to evaluate the prediction accuracy: the root mean square error (RMSE), which is $\text{RMSE} \triangleq \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - a_i)^2}$, the mean absolute error (MAE), which is $\text{MAE} \triangleq \frac{1}{N}\sum_{i=1}^{N}|y_i - a_i|$ and the mean squared logarithmic error (MSLE), which is $\text{MSLE} \triangleq \frac{1}{N}\sum_{i=1}^{N}(\log(y_i+1) - \log(a_i+1))^2$, where N is the number of predicted values, y_i is the forecast value of the i-th sample, and a_i is the actual value of the i-th sample. The lower values of these three metrics mean the better performance of the models. It is worth mentioning that we choose MSLE instead of the mean absolute percentage error (MAPE) because MAPE may produce undefined or infinite value when the actual value is close to zero (or equal to zero) as indicated in [27]. The real traffic flow data often contains the small values close to zero. Thus, MAPE is not suitable for the case.

TABLE I. Performance Comparison with Conventional Models

Models	CPeMS dataset			FBBC dataset		
	RMSE	MAE	MSLE	RMSE	MAE	MSLE
ARIMA	6.25e+01	5.42e+01	1.97e+00	1.44e+02	1.08e+02	2.60e+00
SVR	5.65e+00	4.70e+00	3.33e-01	1.21e+02	7.43e+01	1.40e+00
DR	5.72e-02	4.31e-02	1.60e-03	4.08e-02	2.30e-02	8.80e-04
CNN	4.81e-02	3.71e-02	1.20e-03	3.37e-02	2.07e-02	7.76e-04
SAES	4.85e-02	3.55e-02	1.16e-03	2.44e-02	1.32e-02	3.90e-04
LSTM	4.59e-02	2.82e-02	1.07e-03	2.24e-02	1.21e-02	3.17e-04
GRU	4.43e-02	2.39e-02	1.01e-03	2.17e-02	1.18e-02	3.09e-04
LSTM-CNN	3.78e-02	2.87e-02	7.60e-04	2.82e-02	1.71e-02	5.62e-04
DCN	8.49e-03	5.65e-03	8.97e-05	8.05e-03	4.88e-03	5.39e-05
WADC	**4.19e-03**	**2.60e-03**	**1.09e-05**	**2.51e-03**	**1.23e-03**	**4.80e-06**

B. Experimental Analysis

1) Baseline models: We perform a number of experiments to investigate our WADC model in contrast to other conventional baselines models. In particular, we categorize the following nine representative baseline models into three types: 1) the traditional parametric and ML models; 2) the sole deep learning models; 3) the composite deep learning models.

Traditional parametric and ML models include ARIMA, SVR and DR model.

- *ARIMA* is able to capture a suite of different typical temporal structures in time series data. Moreover, ARIMA is a financial forecasting method that has been frequently utilized in time series investigation of financial data [28].
- *SVR* is an extension of SVM. In addition, SVR is essentially a Support-Vector Classification (SVC), in which a radial basis function kernel is adopted. It is also a basic traditional ML method to support many applications such as stock forecasting [29].
- *DR* is a basical neural network, which is composed of two layers with $\tanh$ activation function in this experiment. It is a typical traditional ML model with different activation functions [30].

Sole deep learning (DL) models include CNN, Stacked auto-encoders (SAES), LSTM and GRU.

- *CNN* is composed of several convolutional layers alternating with pooling layers, and a fully-connected layer. The CNN model shows the strengths in training complex and extensive data. In the experiment, We mainly investigate the CNN structure with 2 convolutional layers.
- *SAES* is built by stacking auto-encoders to construct a deep neural network, proposed in [15].
- *LSTM* can be applicable to the time series features predictions. In our experiment, we conduct the LSTM structure with 2 LSTM layers for prediction tasks.
- *GRU* structure is a gating mechanism in recurrent neural networks, proposed in [31]. Compared with LSTM, GRU has fewer parameters due to the absence of an output gate.

Composite deep learning models contain more than one ML or DL models. We choose the following representative composite models.

- *LSTM-CNN* consists of CNN layers following multiple LSTM layers. We carry out LSTM-CNN model with 2 LSTM layers and 2 convolutional layers to perform the forecasting experiment.
- *DCN* mainly consists of a cross network and deep network, proposed in [21]. DCN is a variant of the wide and deep architecture, as one of the most up-to-date models for prediction and classification tasks.

Our proposed WADC model consists of a wide-attention module and a deep-composite module. This model includes 1 linear model with a self-attention mechanism in wide-attention module, 1 LSTM layer and 1 convolutional layer in deep-composite module. Furthermore, we adopt $L_1\&L_2$ regularization optimization methods for wide-attention module, select 128 convolution filters and 12 LSTM neurons in the deep-composite module.

2) Baseline experimental results: Table I shows the performance comparison of the WADC model with other conventional models. It is worth mentioning that our baseline experiments are conducted on two datasets: CPeMS and FBBC. In each dataset, we evaluate 10 models in terms of three performance metrics, i.e., RMSE, RMAE and MSLE.

- **Analysis of traditional ML approaches:** As shown in Table I, compared with other DL models, traditional ML models such as ARIMA, SVR and DR models have much higher values of three evaluation metrics (including RMSE, MAE and MSLE); this implies the poorer performance of traditional ML models. For example, ARIMA only achieves 6.25e+01, 5.42e+01 and 1.97e+00 in RMSE, MAE and MSLE, respectively. The results are the largest values among all baseline models in CPeMS dataset prediction.
- **Analysis of sole DL approaches:** Compared with ML models, most of these sole DL models achieve the improved results (in terms of lower values of RMSE, MAE and MSLE). The reason may owe to the fact that DL models have superiority in generalization especially after learning extensive traffic flow data. In addition, we also compare 4 sole DL models with each other. As shown in Table I, both GRU and LSTM perform better than CNN and SEAS. For example, GRU achieves 4.43e-02, 2.39e-02 and 1.01e-03 in RMSE, MAE and MSLE, respectively, which is the best among the sole models in CPeMS dataset. This result implies that GRU and LSTM have the advantage in learning time series data since they can preserve long-term memory of features.
- **Analysis of composite DL approaches:** It is shown in Table I that most of the composite DL models outperform traditional ML and sole DL models. Furthermore, compared with other composite DL models such as LSTM-CNN and DCN models, our WADC can achieve even better performance. The performance improvement may owe to the superior learning capability of WADC brought by the wide-attention module and the deep-composite module coordination.

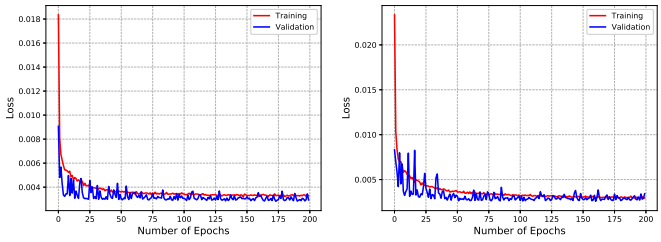
3) Training phase comparison: We then perform extensive experiments to evaluate the performance of different models in the training phase. We select six representative models including DR (achieving the best performance among traditional ML models), all sole DL models (i.e., CNN, SAES, GRU and LSTM) and our WADC model to conduct the following experiments after 200 training iterations (epochs) mainly based on CPeMS and FBBC datasets.

Traditional ML models vs. Sole DL models: We first compare DR model with CNN model. Both DR and CNN models are the most representative baselines in the traditional ML and the sole DL models, respectively. Fig. 3 and Fig. 5 show the comparison of deep regress and CNN in terms of loss. Compared with the loss of CNN, we can observe that DR model has slow convergence and poor performance (in terms of loss) in validation on FBBC dataset. Furthermore, both DR and CNN do not achieve the best results since the loss values of them are still unstable after 200 epochs.

Composite DL models vs. Sole DL models: We then compare the sole DL models with the composite DL model. We select SAES, GRU and LSTM as the sole DL models and WADC as the composite model. Fig. 4 and Fig. 6 indicate that the loss of every model decreases with the increased number of epochs. For example, the loss of every model is relatively stable when the number of epochs is about 50. Moreover, we observe from Fig. 4 and Fig. 6 that both GRU and LSTM have faster convergence speed than SAES. In particular, compared with other sole DL models, the loss of our WADC achieves the fastest convergence and becomes fairly stable after 25 epochs. This result also implies that superior performance of our WADC model.

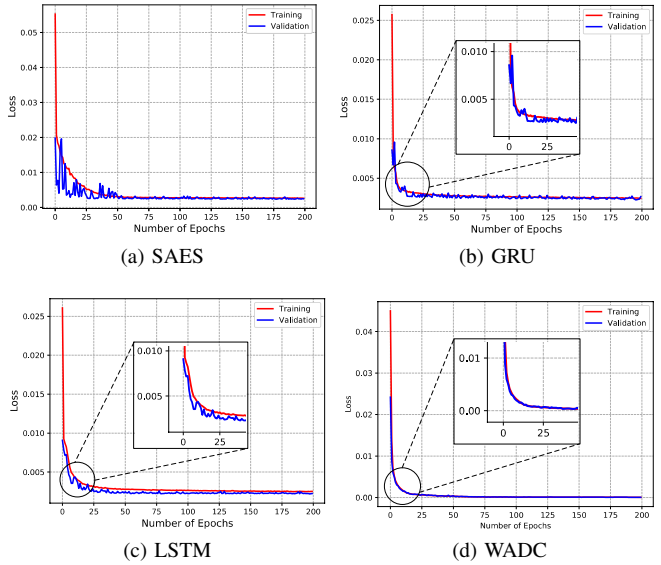
C. Impacts of parameters

Then, we also analyze the performance impacts of different parameters on WADC. As indicated in the above experimental results, different datasets have the little influence on the performance. Therefore, we execute the next group of experiments based on a single dataset (i.e., CPeMS dataset). Specially, we also consider the following key parameters: 1) α denotes regularization methods of attention mechanism; 2) β denotes the number of CNN filters; 3) γ denotes the number of LSTM neurons.



(a) DR (b) CNN

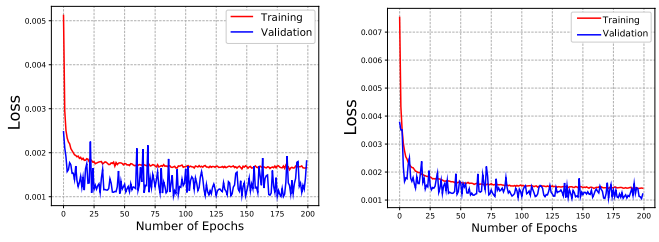
Fig. 3. Traditional ML model vs. Sole DL model on CPeMS dataset



(a) SAES (b) GRU

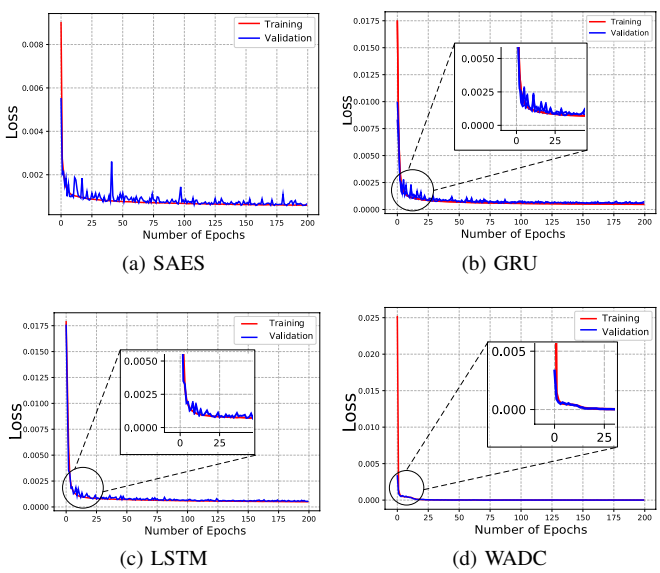
(c) LSTM (d) WADC

Fig. 4. Sole DL model vs. Composite DL model on CPeMS dataset



(a) DR (b) CNN

Fig. 5. Traditional ML model vs. Sole DL model on FBBC dataset



(a) SAES (b) GRU

(c) LSTM (d) WADC

Fig. 6. Sole DL model vs. Composite DL model on FBBC dataset

1) Effect of regularization methods of attention mechanism: We first consider the impact of α. In particular, the regularization methods of attention mechanism include L_1 and L_2 regularizations. Therefore, we define $\alpha \in$ (*None*, L_1, L_2 and $L_1\&L_2$), where *None* denotes WADC without any regularization methods, L_1 and L_2 denote sole L_1 regularization, sole L_2 regularization being utilized, respectively. Specially, WADC can also be optimized by both kernel-regularizer and bias-regularizer, denoted by $L_1\&L_2$. We then vary α according to the following set of values (*None*, L_1, L_2 and $L_1\&L_2$). Meanwhile, we fix β to 128 and γ to 12. We implement two groups of experiments with RMSE and MAE as metrics.

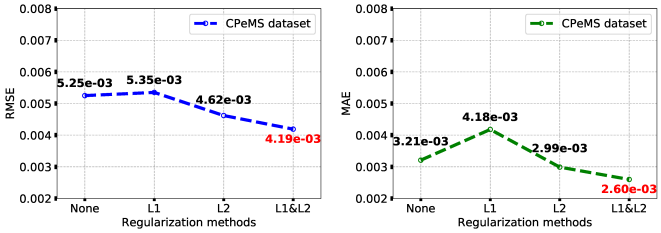
Fig. 7 presents the experimental results after 600 iterations. Compared with $\alpha =$ *None* and $\alpha = L_1$, we can observe that both RMSE and MAE decrease when α is L_2. Moreover, when α is $L_1\&L_2$ (i.e., both L_1 and L_2 regularization methods are used), WADC achieves an even better performance than the model with L_2 regularization only. These results imply the proper adoption of regularization methods has the significant influence on the performance.

2) Effect of No. of CNN Filters: We next consider the influence brought by β in the CNN component of the deep-composite module. Then, we vary β from 32, 64, 128 and 256 and fix α to be $L_1\&L_2$ in the wide-attention module and γ to be 12.

Fig. 8 shows that RMSE increases when β increases from 32 to 64. Both RMSE and MAE decrease when β is varied from 64 to 128. Then, when β is larger than 128, both RMSE and MAE increase with more CNN filters. It implies that the optimal value of β is 128. In particular, we can note that RMSE and MAE have different trends. The reason can be explained by the fact that the loss values are squared before being averaged in RMSE calculation. Therefore, RMSE may give a relatively higher weight to the large loss value as indicated in [32]. Another possible reason may lie in the increased training difficulty with more CNN filters. That is the reason why we adopt both RMSE and MAE to investigate the impacts of parameters on the performance.

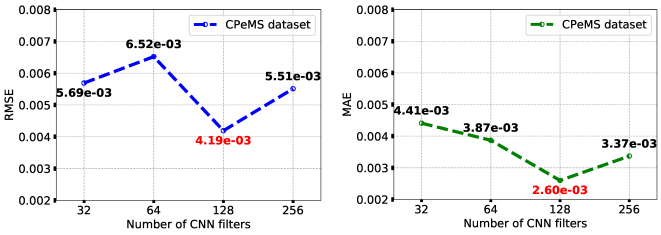
3) Effect of No. of LSTM Neurons: We also consider the impact of γ. We then vary γ from 8, 12, 32 to 64 in the LSTM component. We fix α to be $L_1\&L_2$ regularization methods in the wide-attention module and β to be 128 in CNN component.

Fig. 9 shows the results of two groups of experiments. It is worth noting that both RMSE and MAE first drop with γ



(a) RMSE (b) MAE

Fig. 7. Effect of Regularization Methods of Attention Mechanism



(a) RMSE (b) MAE

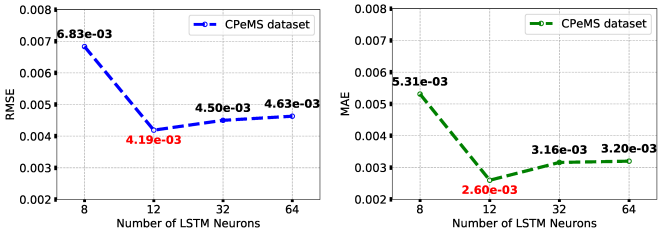
Fig. 8. Effect of Number of CNN Filters

being increased from 8 to 12. When γ is larger than 12, both RMSE and MAE rise with the increment of γ. It reveals that the optimal value of γ is 12.

Table II shows the impacts of parameters comparison for WADC model by three evaluation metrics. As shown in Table II, we can also observe that when $\alpha = L_1\&L_2$, $\beta = 128$ and $\gamma = 12$, RMSE, MAE and MSLE are 4.19e-03, 2.60e-03, and 1.09e-05, respectively. Compared with models with other settings, WADC with $\alpha = L_1\&L_2$, $\beta = 128$ and $\gamma = 12$ achieved the best performance in RMSE, MAE and MSLE.

V. Conclusion

In this paper, we put forth a wide-attention and deep-composite (WADC) model for traffic flow prediction. Specially, our WADC is composed of a wide-attention module and a deep-composite module. The wide-attention module can extract the feature interactions though a linear model with self-attention mechanism. The deep-composite module mainly consists of CNN and LSTM components, which can generalize feature combinations from the traffic flow data. Moreover, we evaluate WADC through conducting performance comparison with other nine existing ML and DL models via extensive experiments. Our experimental results reveal that WADC outperforms traditional ML and DL (including sole DL models and composite DL models) approaches like ARIMA, LSTM and DCN. Regarding future work, we are going to examine the performance advances of our WADC model via regulating various numbers of both CNN and LSTM layers when interpreting various types of traffic flow features. In addition, we will explore the possible application of our traffic flow prediction model to the emerging edge/cloud TCPS [33]. Moreover, we will also investigate the adoption of our model



(a) RMSE (b) MAE

Fig. 9. Effect of Number of LSTM Neurons

TABLE II. Impacts of parameters comparison for WADC model in terms of three evaluation metrics

Parameters	Options	CPeMS dataset		
		RMSE	MAE	MSLE
α ($\beta = 128$, $\gamma = 12$)	None	5.25e-03	3.21e-03	7.81e-05
	L1	5.35e-03	4.18e-03	1.57e-05
	L2	4.62e-03	2.99e-03	1.10e-05
β ($\alpha = L_1\&L_2$, $\gamma = 12$)	32	5.69e-03	4.41e-03	1.90e-05
	64	6.52e-03	3.87e-03	1.89e-05
	256	5.51e-03	3.37e-03	1.28e-05
γ ($\alpha = L_1\&L_2$, $\beta = 128$)	8	6.83e-03	5.31e-03	2.93e-05
	32	4.50e-03	3.16e-03	1.18e-05
	64	4.63e-03	3.20e-03	1.56e-05
$\alpha = L_1\&L_2$ $\beta = 128$ $\gamma = 12$		**4.19e-03**	**2.60e-03**	**1.09e-05**

to other industrial scenarios like industrial network intrusion detection [34], [35].

References

[1] X. Zhou, X. Cai, Y. Bu, X. Zheng, J. Jin, T. H. Luan, and C. Li, "When Road Information Meets Data Mining: Precision Detection for Heading and Width of Roads," *IEEE Access*, vol. 7, pp. 60 399–60 410, 2019.

[2] C. Chen, K. Ota, M. Dong, C. Yu, and H. Jin, "WITM: Intelligent Traffic Monitoring Using Fine-Grained Wireless Signal," *IEEE Transactions on Emerging Topics in Computational Intelligence*, pp. 1–10, 2019.

[3] Y.-S. Jeong, Y.-J. Byon, M. M. Castro-Neto, and S. M. Easa, "Supervised weighting-online learning algorithm for short-term traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1700–1707, 2013.

[4] D. Chen, "Research on traffic flow prediction in the big data environment based on the improved RBF neural network," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 2000–2008, 2017.

[5] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, 2017.

[6] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, and Y. Liu, "Deep learning: A generic approach for extreme condition traffic forecasting," in *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 2017, pp. 777–785.

[7] R. H. Shumway and D. S. Stoffer, "ARIMA models," in *Time series analysis and its applications*. Springer, 2017, pp. 75–163.

[8] P. Zarchan and H. Musoff, *Fundamentals of Kalman filtering: a practical approach*. American Institute of Aeronautics and Astronautics, Inc., 2013.

[9] H. Wang, L. Liu, S. Dong, Z. Qian, and H. Wei, "A novel work zone short-term vehicle-type specific traffic speed prediction model through the hybrid EMD–ARIMA framework," *Transportmetrica B: Transport Dynamics*, vol. 4, no. 3, pp. 159–186, 2016.

[10] Z.-H. Wang, C.-Y. Lu, B. Pu, G.-W. Li, and Z.-J. Guo, "Short-term forecast model of vehicles volume based on ARIMA seasonal model and holt-winters," in *ITM Web of Conferences*, vol. 12, 2017.

[11] S. V. Kumar, "Traffic flow prediction using Kalman filtering technique," *Procedia Engineering*, vol. 187, pp. 582–587, 2017.

[12] B. Sharma, S. Kumar, P. Tiwari, P. Yadav, and M. I. Nezhurina, "ANN based short-term traffic flow forecasting in undivided two lane highway," *Journal of Big Data*, vol. 5, no. 1, p. 48, 2018.

[13] J. Zhou, Y. Lu, H.-N. Dai, H. Wang, and H. Xiao, "Sentiment Analysis of Chinese Microblog Based on Stacked Bidirectional LSTM," *IEEE Access*, vol. 7, pp. 38 856–38 866, 2019.

[14] Z. Zheng, Y. Yang, X. Niu, H.-N. Dai, and Y. Zhou, "Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1606–1615, 2018.

[15] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2014.

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

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

[18] S. Deep and X. Zheng, "Hybrid model featuring cnn and lstm architecture for human activity recognition on smartphone sensor data," in *2019 20th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, 2019, pp. 259–264.

[19] Y. Liu, H. Zheng, X. Feng, and Z. Chen, "Short-term traffic flow prediction with Conv-LSTM," in *2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*. IEEE, 2017, pp. 1–6.

[20] H.-T. Cheng *et al.*, "Wide & deep learning for recommender systems," in *Proceedings of the 1st workshop on deep learning for recommender systems*. ACM, 2016, pp. 7–10.

[21] R. Wang, B. Fu, G. Fu, and M. Wang, "Deep & cross network for ad click predictions," in *Proceedings of ACM ADKDD'17*, 2017.

[22] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2018.

[23] J. Ke, H. Zheng, H. Yang, and X. M. Chen, "Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach," *Transportation Research Part C: Emerging Technologies*, vol. 85, pp. 591–608, 2017.

[24] Z. Zheng, Y. Yang, J. Liu, H.-N. Dai, and Y. Zhang, "Deep and embedded learning approach for traffic flow prediction in urban informatics," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3927–3939, 2019.

[25] C. Cortes, M. Mohri, and A. Rostamizadeh, "l_2 regularization for learning kernels," in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, ser. UAI '09, Arlington, Virginia, USA, 2009, pp. 109–116.

[26] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2016.

[27] S. Kim and H. Kim, "A new metric of absolute percentage error for intermittent demand forecasts," *International Journal of Forecasting*, vol. 32, no. 3, pp. 669–679, 2016.

[28] K. Yunus, T. Thiringer, and P. Chen, "ARIMA-Based Frequency-Decomposed Modeling of Wind Speed Time Series," *IEEE Transactions on Power Systems*, vol. 31, no. 4, pp. 2546–2556, July 2016.

[29] S. Madge and S. Bhatt, "Predicting stock price direction using support vector machines," *Independent work report spring*, 2015.

[30] S. Lathuiliere, P. Mesejo, X. Alameda-Pineda, and R. Horaud, "A comprehensive analysis of deep regression," *IEEE transactions on pattern analysis and machine intelligence*, April 2019.

[31] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, 2014.

[32] C. Jao, *Efficient Decision Support Systems: Practice and Challenges From Current to Future*. BoD–Books on Demand, 2011.

[33] X. Peng, K. Ota, and M. Dong, "Multiattribute-Based Double Auction Toward Resource Allocation in Vehicular Fog Computing," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3094–3103, 2020.

[34] W. Liang, K. Li, J. Long, X. Kui, and A. Y. Zomaya, "An Industrial Network Intrusion Detection Algorithm Based on Multifeature Data Clustering Optimization Model," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2063–2071, 2020.

[35] W. Liang, W. Huang, J. Long, K. Zhang, K. Li, and D. Zhang, "Deep Reinforcement Learning for Resource Protection and Real-time Detection in IoT Environment," *IEEE Internet of Things Journal*, pp. 1–1, 2020.