Complexity

Deep Learning Methods Applied to Complex Big Data Analysis

Lead Guest Editor: Min Xia
Guest Editors: Zhijie Wang, Jun Xu, and Cheng Lu

Deep Learning Methods Applied to Complex Big Data Analysis

Complexity

Deep Learning Methods Applied to Complex Big Data Analysis

Lead Guest Editor: Min Xia
Guest Editors: Zhijie Wang, Jun Xu, and Cheng Lu

Contents

A Spatial-Temporal Self-Attention Network (STSAN) for Location Prediction
Shuang Wang, AnLiang Li, Shuai Xie, WenZhu Li, BoWei Wang, Shuai Yao, and Muhammad Asif
Research Article (13 pages), Article ID 6692313, Volume 2021 (2021)

Learning Air Traffic as Images: A Deep Convolutional Neural Network for Airspace Operation Complexity Evaluation
Hua Xie, Minghua Zhang, Jiaming Ge, Xinfang Dong, and Haiyan Chen
Research Article (16 pages), Article ID 6457246, Volume 2021 (2021)

DefogNet: A Single-Image Dehazing Algorithm with Cyclic Structure and Cross-Layer Connections
Suting Chen, Wenhao Fan, Shaw Peter, Chuang Zhang, Kui Chen, and Yong Huang
Research Article (13 pages), Article ID 2352185, Volume 2021 (2021)

MSLp: Deep Superresolution for Meteorological Satellite Image
Liling Zhao, Hao Yu, and Yan Wang
Research Article (8 pages), Article ID 2678124, Volume 2021 (2021)

Realistic Speech-Driven Talking Video Generation with Personalized Pose
Xu Zhang and Liguo Weng
Research Article (8 pages), Article ID 6629634, Volume 2020 (2020)

A Multi-Index Generative Adversarial Network for Tool Wear Detection with Imbalanced Data
Guokai Zhang, Haoping Xiao, Jingwen Jiang, Qinyuan Liu, Yimo Liu, and Liying Wang
Review Article (10 pages), Article ID 5831632, Volume 2020 (2020)

A Hybrid Prediction Method for Stock Price Using LSTM and Ensemble EMD
Yang Yujun, Yang Yimei, and Xiao Jianhua
Research Article (16 pages), Article ID 6431712, Volume 2020 (2020)

Feature Guided CNN for Baby's Facial Expression Recognition
Qing Lin, Ruili He, and Peihe Jiang
Research Article (10 pages), Article ID 8855885, Volume 2020 (2020)

Improved ML-Based Technique for Credit Card Scoring in Internet Financial Risk Control
Shuangshuang Fan, Yanbo Shen, and Shengnan Peng
Research Article (14 pages), Article ID 8706285, Volume 2020 (2020)

A Methodology for Calculating Greenhouse Effect of Aircraft Cruise Using Genetic Algorithm-Optimized Wavelet Neural Network
Yong Tian, Lina Ma, Songtao Yang, and Qian Wang
Research Article (13 pages), Article ID 7141320, Volume 2020 (2020)

Airport Arrival Flow Prediction considering Meteorological Factors Based on Deep-Learning Methods
Zhao Yang, Yifan Wang, Jie Li, Liming Liu, Jiyang Ma, and Yi Zhong
Research Article (11 pages), Article ID 6309272, Volume 2020 (2020)

A Marine Object Detection Algorithm Based on SSD and Feature Enhancement
Kai Hu, Feiyu Lu, Meixia Lu, Zhiliang Deng, and Yunping Liu
Research Article (14 pages), Article ID 5476142, Volume 2020 (2020)

Prediction for Chaotic Time Series-Based AE-CNN and Transfer Learning
Baogui Xin and Wei Peng
Research Article (9 pages), Article ID 2680480, Volume 2020 (2020)

Hindawi
Complexity
Volume 2021, Article ID 6692313, 13 pages
https://doi.org/10.1155/2021/6692313

Research Article

A Spatial-Temporal Self-Attention Network (STSAN) for Location Prediction

Shuang Wang,[1] **AnLiang Li**,[1] **Shuai Xie**,[1] **WenZhu Li**,[1] **BoWei Wang**,[1] **Shuai Yao**,[1] **and Muhammad Asif**[2]

[1]*School of Software, Northeastern University, Shenyang 110000, China*
[2]*Department of Computer Science, Ekha Ghund Degree College Mohmand, Peshawar, KpK 24650, Pakistan*

Correspondence should be addressed to Shuang Wang; wangsh@mail.neu.edu.cn

Received 17 October 2020; Revised 16 March 2021; Accepted 22 March 2021; Published 5 April 2021

Academic Editor: Min Xia

With the popularity of location-based social networks, location prediction has become an important task and has gained significant attention in recent years. However, how to use massive trajectory data and spatial-temporal context information effectively to mine the user's mobility pattern and predict the users' next location is still unresolved. In this paper, we propose a novel network named STSAN (spatial-temporal self-attention network), which can integrate spatial-temporal information with the self-attention for location prediction. In STSAN, we design a trajectory attention module to learn users' dynamic trajectory representation, which includes three modules: location attention, which captures the location sequential transitions with self-attention; spatial attention, which captures user's preference for geographic location; and temporal attention, which captures the user temporal activity preference. Finally, extensive experiments on four real-world check-ins datasets are designed to verify the effectiveness of our proposed method. Experimental results show that spatial-temporal information can effectively improve the performance of the model. Our method STSAN gains about 39.8% Acc@1 and 4.4% APR improvements against the strongest baseline on New York City dataset.

1. Introduction

Location-based social networks (LBSNs), such as Foursquare and Gowalla, become increasingly popular, and user-generated digital footprints bring an unprecedented opportunity for exploration of the human mobility patterns. Human mobility and mobility patterns have a high degree of freedom and diversity, so capturing human mobility patterns is a challenging task in LBSN applications, such as personalized recommendation and preference-based route planning.

The traditional methods for location prediction leverage Markov chains (MCs) to capture the transition regularities of human movements [1, 2]. However, the transition probability of the model is predefined, and only the impact of the last check-in activity can be considered. Since deep learning technology has a strong ability of representation learning, recently some location prediction work [3] uses embedding methods and recurrent neural networks (RNNs) to learn location embedding and user representation, respectively. Although the performance has been improved, it cannot solve the problem of gradient vanishing in the training process. To solve this problem, scholars have done a lot of research, using the RNN variants (LSTM [4] and GRU [5]) in the encoder-decoder framework. However, for the sequences over 50 in length, LSTM's perception of further check-in activities is weak, and long-term dependence is still difficult to capture [6]. More recently, the attention mechanism is introduced to dynamically adjust the weight of important records, and it partially resolves the problem that LSTM cannot learn long-distance dependency. However, the one-dimensional attention may neutralize the relationship between vectors, which makes it difficult to discard the

uncorrelated parts in the weighted average vector in order to only remain the highly semantically related content [7].

On the contrary, data records have rich contextual information but are sparse. Spatial and temporal contexts are two key factors [3, 8], which can be effectively used to alleviate the data sparsity. Some studies [6, 9] divide time into 48 hours and encode it into a time feature vector but ignore the influence of spatial context. Kong and Wu [8] encode the time and geographic distance interval as vectors and integrate them into the LSTM module as a time gate and geographic gate to jointly consider the spatial-temporal information. However, this method can only consider the information between adjacent trajectory points in the trajectory sequence, and it cannot tackle the spatial-temporal information between any trajectory points globally and ignores the user's geographical location preference.

To overcome the problems mentioned above, we propose a novel network named STSAN (spatial-temporal self-attention network), which can integrate spatial-temporal information with the self-attention [10] mechanism for location prediction. We design a trajectory attention network to learn users' dynamic trajectory representation, which includes three modules: location attention, spatial attention, and temporal attention. Our model can capture the complicated transitions and the user's preference for geographic location and temporal activity. Finally, different from the previous methods of learning user representation, we represent user trajectory by trajectory points, which are represented in the location embedding space. Our model can avoid compressing user trajectory into a vector, reducing the loss of trajectory information.

The major contributions of this paper can be summarized as follows:

(i) We propose a novel network, STSAN, to capture complex sequential transition regularities and integrate spatial-temporal information for location prediction. Our model can capture the time interval information between any two points in the trajectory and can also sense the user's geographical preference.

(ii) We propose that the trajectory is represented in location latent vector space by the trajectory points instead of the user representation, which avoids the compression and loss of the trajectory information.

(iii) We experimentally evaluate our model using four datasets collected from Foursquare and Gowalla. The experiment results show the superiority of our approach over various baseline approaches, and it is more prominent in sparse datasets. Our model achieves 39.8% Acc@1 and 4.4% APR over VANext [11], which performs the second best on New York City dataset.

2. Related Work

2.1. Location Prediction. Extensive studies have been dedicated to model human mobility via large-scale trajectory data recorded by GPS, cellular towers, and location-based service. The traditional methods are based on the Markov chains (MCs) model which represents the individual's movement behavior as a Markov model. The MCs calculate a state (location) transition matrix and predict the next location based on the previously visited location [2, 10]. Although time-series information is considered, only a short-range time-series relationship is modeled, which limits its prediction ability. With the widespread development of deep learning research, many neural network models are used to discover the user's mobility patterns. The PRME [12] algorithm embedded users and locations into the hidden space to explore a similar relationship, but it can only model the short-range relationship of the sequence. Recently, recurrent neural networks and their variants have been widely used to capture long-term sequence effects. The STRNN [3] used the RNN model combined with temporal and spatial contextual information to predict the next location. The recurrent neural network model based on spatial-temporal features [13] can automatically extract the internal representation of spatial and temporal features and combine RNN structure for modeling people's movement behavior. It is very difficult to deal with long sequence data in the RNN model, so recently RNN variants (LSTM and GRU) have been proposed to avoid the gradient disappearance of conventional RNN. The HST-LSTM model [8] embedded time and space vectors into LSTM in the framework of encoder and decoder to predict the next location of users. With the further development of deep learning technology, an attention mechanism has been proposed to enhance the learning of RNN and CNN structures on the long-term dependence of longer sequences. The DeepMove [6] used the attention mechanism to enhance RNN to capture the user's mobility and location preference. The VANext [11] used CNN and attention to learn the periodical patterns of historical trajectory to make the next location prediction. The AMF [14] combined a personalized federated learning model with an attention network for location prediction.

2.2. Attention. The attention mechanism is widely used in classify images [15], machine translation [16], and various NLP tasks [17–19]. In 2017, the Google teams propose a self-attention model [10] to learn text representation, and self-attention outperforms RNN model with an attention mechanism in sequence to sequence task. Since then, self-attention has been widely used in the recommendation system. In 2017, ATRank [7] used self-attention to capture the impact of users' different behaviors, model user behaviors, and apply it to downstream recommendation tasks. In 2018, Zhang et al. [20] used self-attention to learn the relationship between items and items in user history interaction and made the next item recommendation. In 2018, self-attention is used as a sequence recommendation model, and self-attention is used to capture long-term semantic sequences and attention makes its predictions based on relatively few actions [21]. Xia et al. [22, 23] applied attention in satellite image segmentation to extract useful information and ignore useless information. A global attention module was used to weight low-level features in water segmentation

task. MFANet [24] used a multilevel feature attention module to provide information for the low-level features by using high-level features to generate new features in the segmentation of remote sensing images.

The standard self-attention can only deal with some simple sequence data, such as sentence sequence or timeseries, but it cannot deal with more complex trajectory data for the location prediction problem because trajectory contains timestamps, geographical location coordinate, and other heterogeneous contexts. We propose a trajectory attention mechanism, which integrates geographic and temporal features based on standard self-attention to learn user trajectory representation. Trajectory attention includes three modules: location attention, which learns the relevance between the location vectors in the trajectory; spatial attention, which uses spatial features to learn the user's preference for spatial location; and temporal attention, which captures different time preferences among users.

3. Preliminaries

In this section, we first introduce some concepts that are necessary for subsequent discussion and then give an overview of the issues discussed in this paper. Finally, a brief introduction of our model framework is given.

3.1. Problem Formulation. Let U and V denote the sets of users and POIs and $|U|$ and $|V|$ represent the number of users and POIs, respectively.

Definition 1 (POI). Point of interest (POI) is a location in a coordinate system, which contains location identification v and the geographical position information (latitude and longitude coordinates).

Definition 2 (trajectory point). A check-in record contains the user identification u, POI v, and check-in timestamp t. The user u visited POI v at t is defined as a tuple $\mathrm{pt}^u = \langle v, t\rangle$, which is also called a user's trajectory point.

Definition 3 (trajectory). Given all trajectory points of the user u, the trajectory is a sequence $\mathrm{traj}^u = \mathrm{pt}_1, \ldots, \mathrm{pt}_{L^u}$, where L^u is the length of the user's trajectory sequence and pt_i is the i-th trajectory point of the user u.

Location prediction problem. Given the set U of users and the set V of POIs and the current trajectory sequence traj^u for a user $u \in U$, the problem is to predict the location $v_{L^u+1} \in V$ in the $(L^u + 1)$-th timestamp of user u.

3.2. Basic Framework. Figure 1 shows the architecture of STSAN, which consists of three major components: (1) trajectory feature processing, (2) trajectory attention module, and (3) prediction.

(i) Trajectory feature processing: we first embed each POI into a low-dimensional vector representation using an embedding method. To build a spatial-temporal user trajectory prediction model, the timestamp and the POI's coordinates are taken as the numerical characteristics of the model.

(ii) Trajectory attention module: we use trajectory attention for learning trajectory representation. Specifically, the location attention calculates the relevance between the location vectors in the trajectory, to capture user's mobility patterns; then, the temporal attention uses the time feature to calculate the time correlation between the trajectory points and to capture the user's mobility patterns. The spatial attention uses spatial features to learn the spatial relationship between POIs and capture the geographic preference of users for POIs. We use the linear function to integrate three output matrices of attention module. Finally, the output of this module is trajectory representation.

(iii) Prediction: the trajectory representation is represented by trajectory points, which are represented in the location embedding space. The inner product is used to calculate the correlation between the trajectory and the location and then infer the user's next visit POI.

4. Methodology

4.1. Trajectory Feature Processing. To build a spatial-temporal user preference model, we would like to encode the trajectory according to the spatial and temporal characteristics of the user's activities. Word2vec [25] is an effective and scalable method to learn embedding representations by modeling words' contextual correlations in word sentences. We encode each location identification into a low-dimensional vector. For each location, the embedding method outputs the embedded feature vector $\mathbf{p_i}$. The formula representation is as follows:

$$\mathbf{p_i} = a_i W_p, \tag{1}$$

where a_i is the one-hot representation of the i-th candidate location and $W_p \in \mathbb{R}^{|V|\times d_p}$ is the location embedding matrix, which is trained and learned by the whole network, where $|V|$ is the number of candidate sites and d_p is the dimension of the embedding feature vector.

Previous work [8] has shown that temporal and spatial features help capture user check-in activities. We extract the timestamp t and latitude and longitude of the place where the user visits and use the location embedding vector as the input of the model. For a user trajectory $\mathrm{traj}^u = \mathrm{point}_1, \mathrm{point}_2, \ldots, \mathrm{point}_L$, we set $L^u = L$, and the input of the model is

$$X^u = x_1, x_2, \ldots, x_L, \tag{2}$$

where $x_i = \{\mathbf{p}_i, (\mathrm{latitude}, \mathrm{longitude})_i, t_i\}$ contains three parts: $\mathbf{p_i}$ is the embedding vector of the i-th location in the trajectory, $(\mathrm{latitude}, \mathrm{longitude})_i$ is the latitude and longitude for this location, and t_i is the timestamp of the visited location.

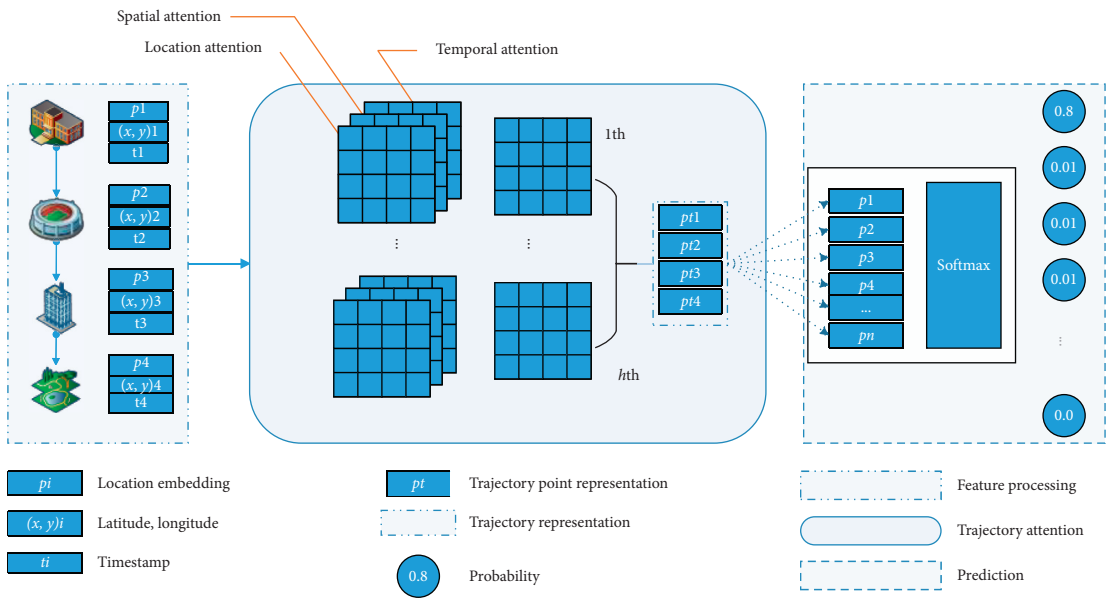


Figure 1: The overall architecture of STSAN. First the feature information is extracted from the entire history of a user by feature extraction module. Then, we apply trajectory model to obtain user trajectory representation. At the end of the sequence, the hidden state of trajectory attention is passed through a softmax layer for next location prediction.

4.2. Trajectory Attention Module. Figure 2 shows the architecture of trajectory attention module, which consists of three major components: (1) location attention, (2) temporal attention, and (3) spatial attention.

4.2.1. Location Attention. The self-attention is a special case of attention function that can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. Based on the RNN sequence model, it is not easy to control the long-distance hidden state. The attention mechanism allocates weight to the hidden state and gives more weight to the more important state as much as possible. However, it is still limited by RNN recursive structure and cannot process longer sequences. The self-attention can capture the relationship between arbitrary elements by using sequence information regardless of the position in the sequence. The self-attention can process longer sequences, so we use self-attention to capture the relationship between location embedding vectors.

Suppose the length of the user's trajectory is L. The input of location attention is a matrix in shape of L by d_p:

$$P^u = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_L]^T. \tag{3}$$

Note that the trajectory length is different for different users, and the model can accept trajectory sequence input of different lengths.

To get query, key, and value for user u, we project P^u to three spaces through nonlinear transformation, as shown in the following formulas:

$$\text{Query} = \text{ReLU}\left(P^u W^Q\right), \tag{4}$$

$$\text{Key} = \text{ReLU}\left(P^u W^K\right), \tag{5}$$

$$\text{Value} = \text{ReLU}\left(P^u W^V\right), \tag{6}$$

where $W_Q \in \mathbb{R}^{d \times d_Q}$, $W_K \in \mathbb{R}^{d \times d_K}$, and $W_V \in \mathbb{R}^{d \times d_V}$ are weight matrices for Query, Key, and Value, respectively. ReLU is a nonlinear activation function, which makes the neural network have enough capacities to capture complex patterns. Note that Query, Key, and Value are projected into the same space; hence, $d_Q = d_K = d_V$ and $d = d_p$ can be known by the combination of matrix multiplication in mathematics.

Then, we can calculate the location's attention score using the following formula:

$$S_p^u = \text{soft max}\left(\frac{\text{Query} \cdot \text{Key}^T}{\sqrt{d_K}}\right). \tag{7}$$

The output is the location's attention matrix in shape of L by L. We compute the dot products of the Query and Key, which aims to calculate the similarity or correlation between elements and then divide by $\sqrt{d_K}$ to prevent correlation from being weakened by soft max functions and apply softmax function to get normalized attention weights.

For this module, given a user's check-in location sequence embedding representation P^u, the module output is a weight matrix S_p^u of the similarity or correlation of each element. By learning the sequence of check-in locations of all users, the module captures the common mobile pattern among users and the unique interest preferences of different users. This process is learned by training and adjusting the parameters of the model.

4.2.2. Spatial Attention. We use spatial features of locations to enhance learning about user mobility patterns and user geographic preferences. Specifically, to reduce the repetition

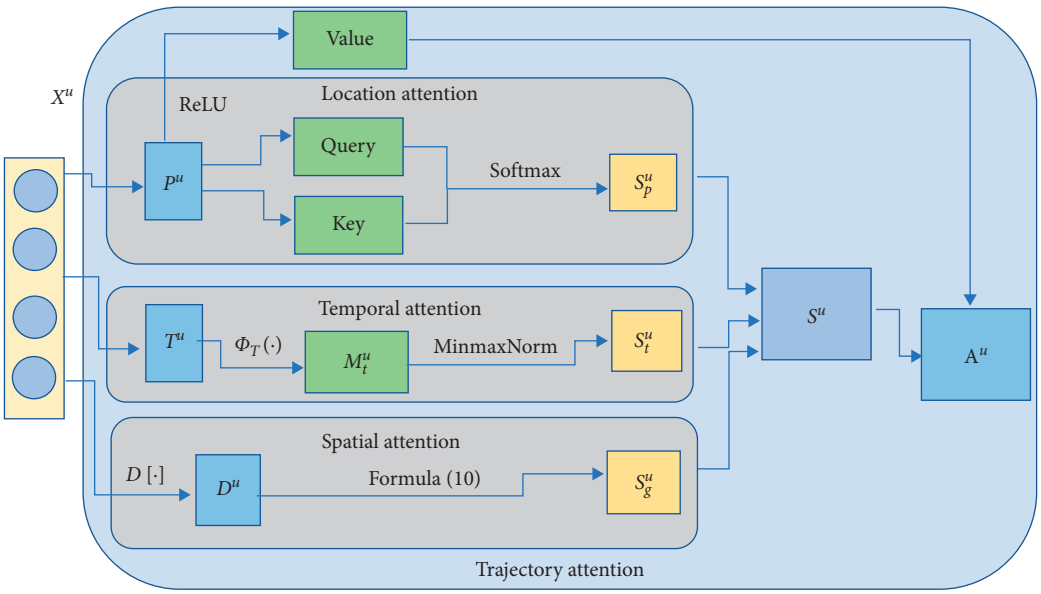


Figure 2: The overall architecture of trajectory attention module.

of calculation, we calculate the geographic distance between all POIs in advance and store it in matrix D and get it by index when using. Given the longitude and latitude of any two POIs of v_i and v_j, the spatial distance between the two POIs $d(v_i, v_j)$ is calculated by using the following formula:

$$\Psi\left(\frac{d(v_i, v_j)}{R}\right) = \Psi(\varphi_2 - \varphi_1) + \cos(\varphi_1)\cos(\varphi_2)\Psi(\Delta\lambda), \tag{8}$$

where $\Psi(\theta) = (1 - \cos(\theta))/2$; R is the radius of the Earth, averaging 6371 km; φ_1 and φ_2 represent the latitude of the two locations; and $\Delta\lambda$ is the longitude difference between the two locations. We set an index operation $D[\cdot]$ that selects the corresponding values of D to form an L by L matrix:

$$D^u = D[X^u]. \tag{9}$$

Note that the value D_{ij}^u in row i and column j of output D^u represents the actual distance between v_i and v_j.

Previous studies [26] have shown that the user's check-in behavior has cluster properties in space. Rather than visiting the places far away, users are more likely to visit the surrounding places that they have gone to, which is congenial with reason and common sense—in our daily life, everyone has their scope of activities, rarely exceeded. To capture this pattern in space, we design spatial attention. To reduce the complexity of the model and facilitate the adjustment of parameters, we manually process the geographical features instead of embedding them into vectors. We set the spatial attention matrix S_g^u in shape of L by L as follows:

$$S_g^u(i, j) = \begin{cases} 1, i \neq j, & D_{ij}^u < \varepsilon, \\ 0, & \text{otherwise}. \end{cases} \tag{10}$$

The purpose of our work is to increase the spatial relevance of locations among users' access areas and indirectly reduce the spatial relevance of locations far away from user preference areas. Note that the distance from a place to itself is zero, so we treat its spatial relevance as a special value of 0. If the value of D_{ij}^u is less than the threshold value ε, the spatial correlation is 1; in other cases, the value is set to 0, and the value on the diagonal is also 0.

4.2.3. Temporal Attention. Different user's check-in behavior has its characteristics in time, such as the time interval. Therefore, we take the two check-in time intervals in the user trajectory as features to capture the time preferences between different users. Similar to spatial attention, we take the time interval between the trajectory points as the feature to calculate the attention in time. The temporal attention input is T^u, that is, a time sequence as follows:

$$T^u = [t_1, t_2, \ldots, t_L]^T. \tag{11}$$

We set the temporal attention matrix S_t^u in shape of L by L as shown in the following formulas:

$$S_t^u = \text{MinMaxNorm}(M_t^u), \tag{12}$$

$$M_t^u = \Phi_T(T^u), \tag{13}$$

$$M_t^u(i, j) = t_i - t_j, \tag{14}$$

where M_t^u is a matrix in shape of L by L and $M_t^u(i, j)$ is the time interval between the check-in of user u at location v_i and location v_j. MinMaxNorm $(\cdot)$ is min-max normalization function that maps the value of the feature to the interval $[0, 1]$. $\Phi_T(\cdot)$ is that we define an operation to map T^u to M_t^u.

We use the parameters β and γ to connect location attention matrix and spatial matrix and temporal matrix linearly and get the total attention matrix formulate as

$$S^u = S_p^u + \beta S_g^u + \gamma S_t^u. \tag{15}$$

The output S^u is a matrix in shape of L by L, representing the location vector relevance affected by time and space features. Hyperparameters β and γ are the time and space influencing factors, indicating the influencing degree of the two factors.

Next, the soft max function is introduced to convert the score S^u of total attention, which is the weight coefficient corresponding to the value, and then the weighted sum is performed:

$$A^u = \text{soft max}\left(S^u\right) \cdot \text{Value}. \tag{16}$$

The output A^u is a matrix in shape of L by d_V. On the one hand, the soft max function can be normalized to sort the original calculated score into a probability distribution with the sum of all element weights of 1; on the other hand, it can also highlight the weight of important elements through the internal mechanism of soft max.

We design h-times attention in different h-spaces, which is similar to the convolution kernel of the convolutional neural network. This allows the model to learn relevant information in different representation subspaces. The results we obtained are as follows:

$$A^u_{K_1}, A^u_{K_2}, \ldots, A^u_{K_h}. \tag{17}$$

Finally, we concatenate the h-order attention results and use the value obtained by linear transformation as the result of multiple attentions:

$$\text{Traj}^u = \text{concat}\left(A^u_{K_1}, A^u_{K_2}, \ldots, A^u_{K_h}\right)W^A, \tag{18}$$

where $W_A \in \mathbb{R}^{hd_V \times d}$ is the projection parameter matrix, the concat $(\cdot)$ connection function concatenates the last dimension of tensors, and the output Traj^u is a matrix in shape of L by d, representing the embedded trajectory for user u. Note that h is the number of parallel attention layers and $d_V = d/h$.

4.3. Prediction. Given the user u_i trajectory sequence traj^u, we use the trajectory attention model to model the embedding vector of trajectory at a higher level to obtain Traj^{u_i}. Specifically, we first map the one-hot vector of locations to the vector space of location through the embedding matrix W_p. In the process of trajectory coding, the trajectory vector is mapped to locations vector space in time order, and then, the closest location to the projection vector is found in the location vector space as the prediction result. We formulate it as

$$p\left(v^{u_i}_{L^u+1} | \text{Traj}^{u_i}\right) = \text{soft max}\left(\text{Traj}^{u_i} W_p^T + b\right). \tag{19}$$

Here, the output $p(v^{u_i}_{L^u+1}|\text{Traj}^{u_i})$ is a probability distribution, which represents the probability distribution of the next position under the condition of known Traj^{u_i} and b is the bias parameter of the current layer network.

Different from the previous work, we not only realize the weight sharing of W_p but also use trajectory points to dynamically represent user trajectory. This method not only reduces the risk of overfitting but also avoids compressing the trajectory to be represented as a vector and keeps more trajectory information.

4.4. Training Algorithm. In order to model user spatial activity preference in a continuous manner, we propose trajectory attention model by considering the spatial and temporal features. In this paper, we consider the next location prediction problem as a multiclassification problem, so we use cross-entropy loss function combining regularization term as the objective function of model training. The objective function of the proposed model is shown as

$$\mathscr{L} = -\sum_{u_i \in U} \sum_{m=1}^{L^{u_i}-1} p^{u_i}_{m+1} \log\left(p\left(v^{u_i}_{m+1} | \text{Traj}^{u_i}_m\right)\right) + \lambda_1 \|\Theta\|_1 + \lambda_2 \|\Theta\|_2, \tag{20}$$

where $\text{Traj}^{u_i}_m$ is trajectory representation at the m-th moment; λ_1, λ_2 are the parameters of $L1$ regularization and $L2$ regularization, respectively; and Θ represents the parameters to be regularized in the model. The gradient descent and backpropagation algorithms are used to modify the network connection parameters, and then, the objective function is minimized. The source code of our model is available online (https://github.com/li-neu/SASAN).

5. Experiments

5.1. Dataset. We evaluate the proposed model with state-of-the-art methods on four check-ins datasets from the following two publicly available datasets:

(i) Foursquare dataset [27]. This dataset contains check-ins in NYC and Tokyo collected for about 10 months (from 12 April 2012 to 16 February 2013). It contains 227,428 check-ins in New York City and 573,703 check-ins in Tokyo. Each check-in is associated with its timestamp and its GPS coordinates.

(ii) Gowalla dataset [28]. This dataset is collected for about 18 months (from 1 February 2009 to 31 October 2010). It contains 6,442,890 check-ins with its timestamp and its GPS coordinates. We select the check-ins in Los Angeles and Houston as experiment dataset.

For each city dataset, we remove users with less than 10 check-in records. The user trajectory is divided into several subtrajectories according to the interval between two adjacent records, and the interval is set to 72 hours, as it was done in previous related works [6]. Then, we remove subtrajectories whose length is less than 2 and remove users whose number of subtrajectories is less than 5. Table 1 shows the basic information of data preprocessing. We calculated the average number of records per user per day on each dataset as the sparsity of the dataset, as shown in Table 1. The size of the time dimension of a single input sample is 5.17, 4.76, 23.33, and 18.94 days, respectively. The maximum length of the model input is set to 50 on all the four datasets. The actual input is not fixed and is determined by the length of the input trajectory.

Table 1: Statistics of datasets.

	Foursquare		Gowalla	
City	New York	Tokyo	Los Angeles	Houston
Users	1,082	2,293	1,057	821
Locations	34,440	56,106	10,657	10,282
Records	184,509	449,640	46,397	45,553
Loc./user	32	24	10	12
Sparsity	0.5684	0.6536	0.0813	0.1027

To show the check-in situation of users in the dataset more intuitively, we draw the cumulative distribution of the check-in quantity over four datasets, which is shown in Figure 3. We can see that the cumulative distribution of the check-in frequency of four datasets is similar to power-law distribution [26, 29]. It can also be seen that the distribution of two cities collected on the same platform is closer, for example, New York and Tokyo collected from Foursquare and Los Angeles and Boston collected from Gowalla. The New York and Tokyo datasets from Foursquare and the Los Angeles and Houston datasets from Gowalla have more users with less than 100 check-ins. About 90% of users in four city datasets have less than 600 check-ins frequency.

5.2. Baselines. To demonstrate the effectiveness of our model, we compare to the following location prediction methods:

(i) STRNN [3]. It is an extended RNN method that can model local temporal and spatial contexts.

(ii) DeepMove [6]. It is a recent location prediction method to learn user periodical patterns with attention mechanism and the recurrent neural networks.

(iii) VANext [11]. It is a location prediction method for variational attention mechanism and leverage CNN to learn historical mobility RNN to learn recent check-in preference.

(iv) Flashback [30]. It is a general RNN architecture design for modeling sparse user mobility traces by doing flashbacks on hidden states in RNNs.

(v) STSAN is a model we proposed, which can integrate spatial-temporal information with the self-attention for location prediction.

5.3. Evaluation Metric. Given the trajectory of the user before the current time, location prediction aims at predicting the next location of a user. Intuitively, a good model is to be able to restore the actual ground records more realistically. Hence, we use the prediction accuracy [6, 31] to measure the performance.

Formally, given the user set **U** and the candidate locations set **C**, for each user, and given the initial location v_1, we predicted the next continuous $L^u - 1$ locations. The prediction accuracy is calculated as follows:

$$\mathrm{Acc@}k = \frac{1}{|U|} \sum_{u_i \in U} \frac{1}{L^{u_i} - 1} \sum_{t=2}^{L^{u_i}} \left(v_t \in S_t^{u_i}(k)\right), \tag{21}$$

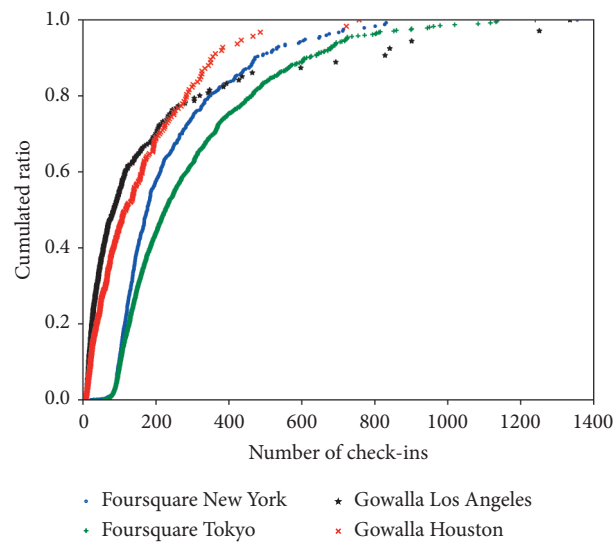


Figure 3: Cumulative distribution function of check-in number.

where $S_t^{u_i}(k)$ is a set of top-k locations for user u_i in time step t, v_t is real visited location for user u_i in time step t, and L^{u_i} is the history trajectory length for user u_i. When v_t in the set $S_t^{u_i}(k)$, the expression value is 1; otherwise, it is 0.

Moreover, similar to the previous work, we apply average percentage rank (APR) [27, 32] as another metric to measure the overall ranking performance of the model. The average percentage rank (APR) is calculated as follows:

$$\mathrm{APR} = \frac{1}{|U|} \sum_{u_i \in U} \frac{1}{L^{u_i} - 1} \sum_{t=2}^{L^{u_i}} \frac{|V| - \mathrm{rank}^{u_i}(v_t) + 1}{|V|}, \tag{22}$$

where $\mathrm{rank}^{u_i}(v_t)$ denotes the rank of candidate location v_t for user u_i in timestamp t after sorting $|V|$ locations in decreasing order.

5.4. Implementation Details. We partition each dataset into a training set and a test set. For each user, we use the first 80% subtrajectory as the training data and the remaining 20% as the test data. We implement our model with PyTorch. All experiments are conducted on an NVIDIA Tesla P4 GPU and a 64G CPU on the Ubuntu system. We use RMSProp adaptive learning rate optimization algorithm [33] and the Reduce LR On Plateau learning rate adaptation method. We clip the $L2$ norm of vector composed of several parameters of gradient to alleviate the problem of gradient explosion. To prevent overfitting, we use $L1$ regularization, weight decay, and dropout [34] to improve the performance of the neural network by preventing the interaction of feature detectors. For the objective function, we set the weighting factors $\lambda_1 = 1e-4$, $\lambda_2 = 1e-5$, dropout rate 0.5, the initial learning rate to $2e-5$, the embedding dimension of location $d_p = 512$, the hidden size $d = 512$, distance threshold $\varepsilon = 20$, spatial factor $\beta = 1.0$, temporal factor $\gamma = 1.0$, the decay of learning rate 0.1, gradient clip 1.0, and the max epoch 50.

5.5. Performance Comparison. We evaluate our model with the baseline methods on four city check-ins datasets to present the performance of our model. Different parameter settings do have an impact on the final performance of the model. Table 2 shows the basic parameter settings of the STSAN model on four datasets when compared with the baselines, and the performance comparison is shown in Figure 4.

Our proposed model STSAN achieves the best performance on four datasets with all evaluation metrics, which illustrates the superiority of our model. Take the New York City dataset; for example, STSAN achieves 39.8% on Acc@1, 51.6% on Acc@5, 53.3% on Acc@10, and 4.4% on APR over VANext [11], which performs the second best. STRNN is an RNN model which incorporates spatial-temporal context. However, these methods do not explicitly model user's historical visit patterns. STRNN has employed recurrent networks to capture long-term POI dependency, and it cannot deal with very long history trajectories well because of the inherent gradient disappearance problem of recurrent networks. Therefore, the main advantage of DeepMove is that it leverages attention network on historical trajectories. Previous techniques try to improve RNNs by considering spatial-temporal context. The context-parameterized transition matrices or gates are used to fuse the spatial-temporal context to simplify the temporal periodicity and spatial law. Flashback [30] is to model the sparse user movement trajectory by flashbacking the hidden state in RNN. VANext not only effectively captures short-term human mobility patterns but also leverages variable attention and CNN networks to generate better historical trajectory representation. It significantly outperforms these previous methods. Therefore, it achieves the best performance than all previous methods. From the results, the performance of the two more sparse datasets (Los Angeles and Houston) is significantly worse. The reason is the model may not be able to get enough data to train, which leads to overfitting problem. For our model STSAN, the performance on sparse datasets has achieved better results on Acc@1. From the latter analysis, we know that the spatial attention module of the model plays a great role.

5.6. Impacts of Spatial and Temporal Attention. To study the influence of temporal and spatial characteristics on the framework, we split the model into three variants: (1) spatial self-attention network (SSAN), which is the network structure without considering the temporal attention module; (2) temporal self-attention network (TSAN), which is the network structure without considering the spatial attention module; (3) self-attention network (SAN), which is the network structure without considering the spatial attention and temporal attention module.

Both Tokyo and New York from foursquare perform similarly, and Houston and Los Angeles from Gowalla perform similarly. Therefore, limited to the length of the article, the article only gives two cities as representatives. Figure 5 shows the performance of STSAN and three variants on the New York and Los Angeles datasets. For the New York dataset, we can see that, with the increase in the epoch, the performance of the four models keeps improving, and after about 10 epochs, they tend to be stable. The performance of TSAN leads SSAN and SAN, so it can be seen that time information is more important than space information on the New York dataset. What is more, we are surprised that STSAN and SSAN both add spatial attention based on the original model, but STSAN brings more performance improvement. When time information and space information are added to the model at the same time, the result of one plus one is greater than two. Finally, we can get the exact conclusion that the performance is STSAN > TSAN > SSAN > SAN on the New York dataset. For the Los Angeles dataset, SSAN leads TSAN and SAN in performance, but TSAN converges faster. Finally, the performance is STSAN > SSAN > TSAN > SAN on the Los Angeles dataset. For different city datasets, spatial and temporal information is not the same for the performance of the model. The reason may be that the spatial information is too complex and not captured well for the New York dataset with more candidate locations.

Table 2: Parameters setting on four datasets.

	γ	β	ε	H	d
New York	1.0	1.0	40	8	512
Tokyo	1.0	1.0	20	32	768
Los Angeles	0.8	0.8	50	2	704
Houston	0.8	0.9	60	1	640

The STSAN has achieved the best performance on all datasets, which proves the importance of using spatial-temporal information modeling. On the one hand, in sparse datasets, spatial factors are more important for model performance improvement. On the other hand, the time factor can help the model converge quickly.

5.7. The Effect of Hyperparameter Settings

5.7.1. Effect of ε and β. The distance threshold ε controls the influence of geographic distance on our model. In general, Figure 6(a) shows that the performance of the model on the four datasets has improved with the increase in distance and the performance improvement on sparse datasets (Los Angeles and Houston) is more significant. We can see that users in different cities have different preferences for distance. Under the default parameter settings, it is more suitable to set ε to 30 and 60 on Los Angeles and Houston datasets, respectively. The spatial factor β controls the influence degree of geographic information on our model. We can see that the accuracy increases in different degrees with the growth of β in the four datasets. The spatial geographic location information is very helpful to improve the accuracy of the model. We also successfully capture it with spatial attention.

5.7.2. Effect of γ. Figure 6(b) shows the APR and Acc@10 for various γ that control the influence of degree of temporal information while keeping other optimal hyperparameters

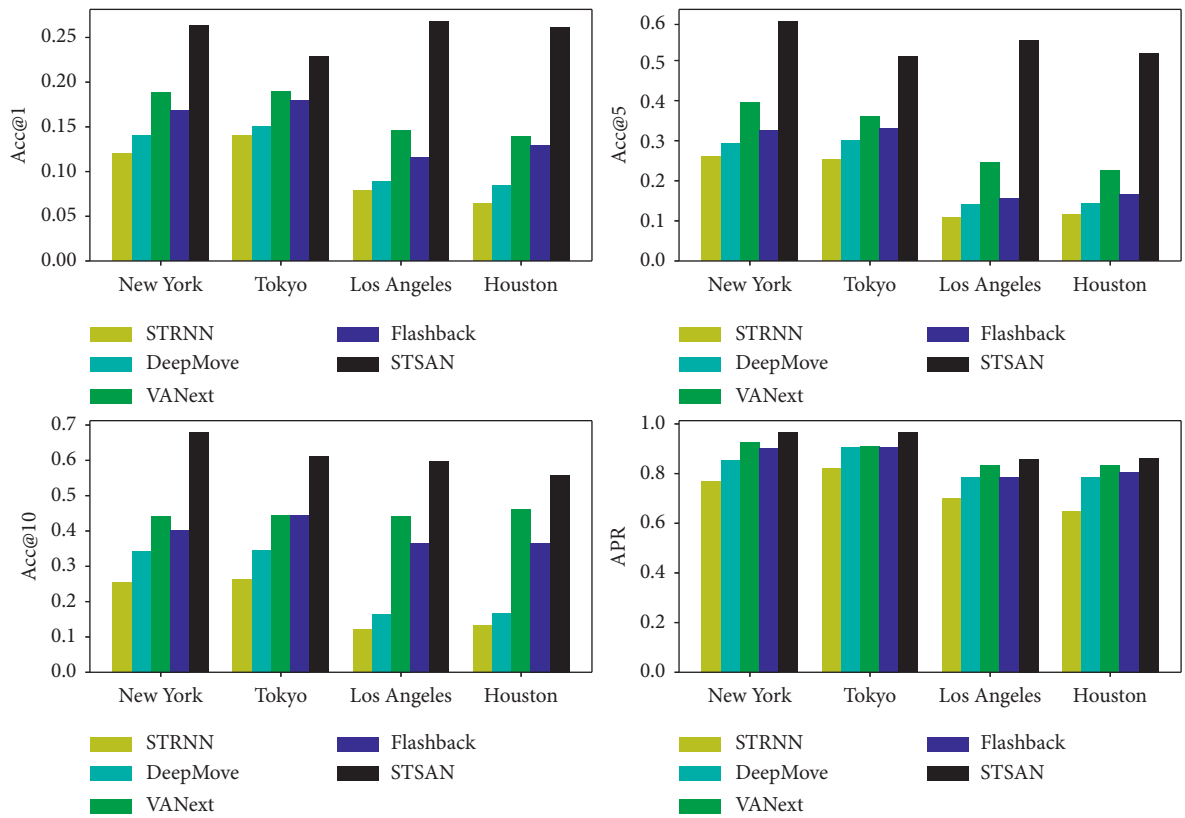


Figure 4: Performance compared with the baselines on the four datasets.

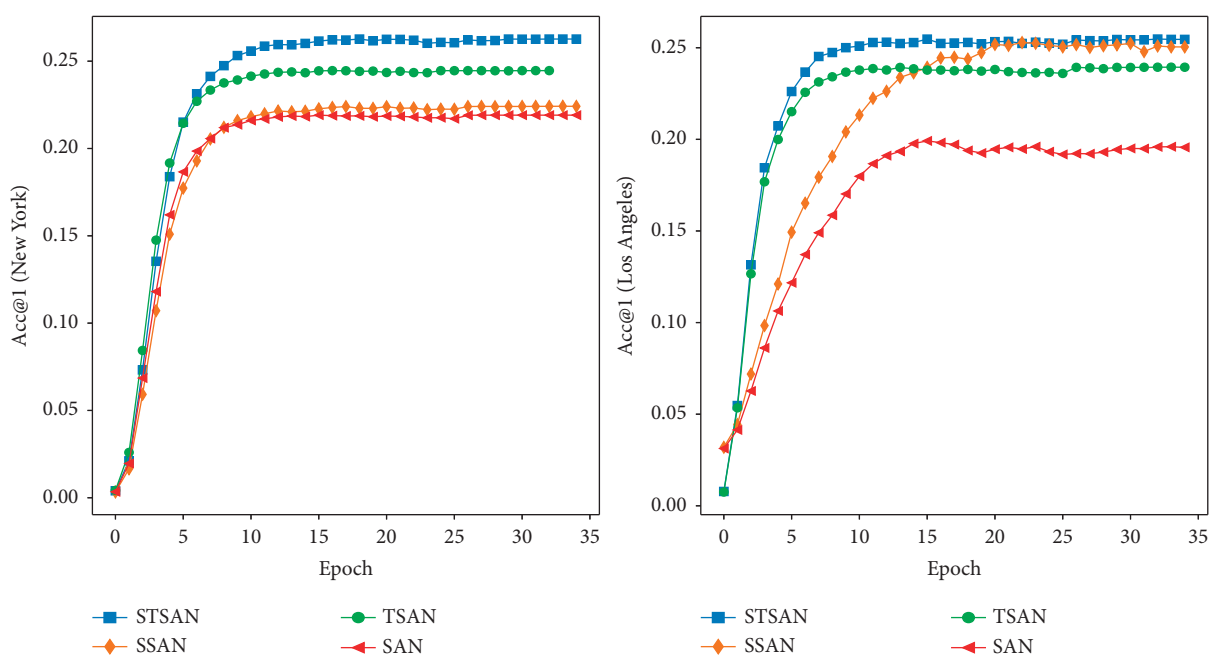
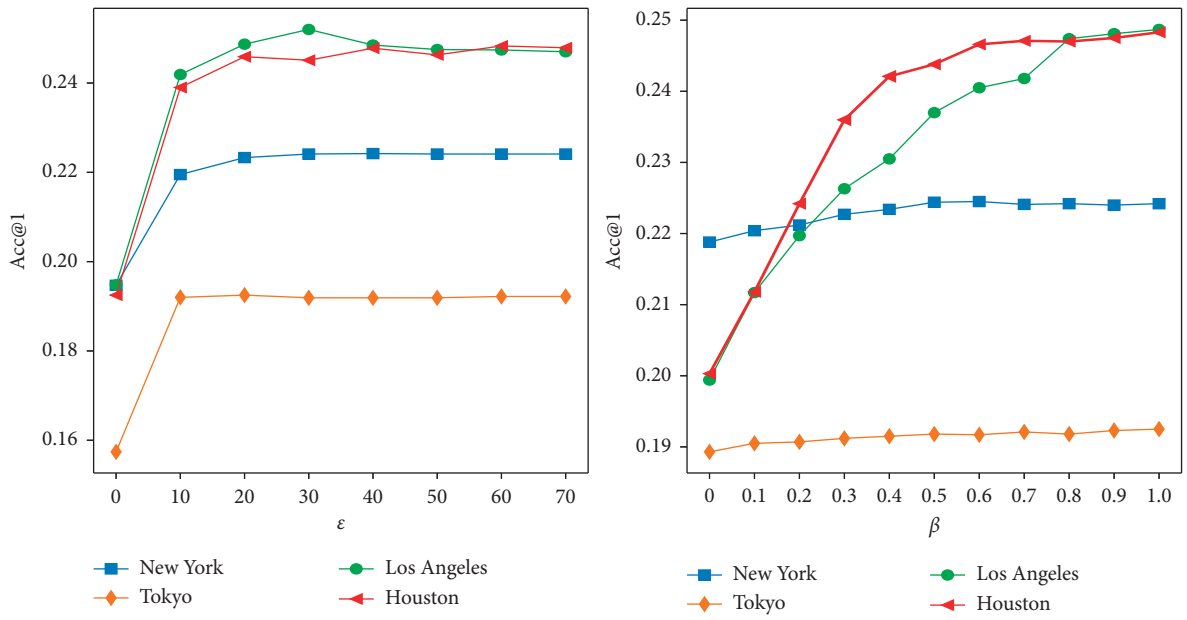
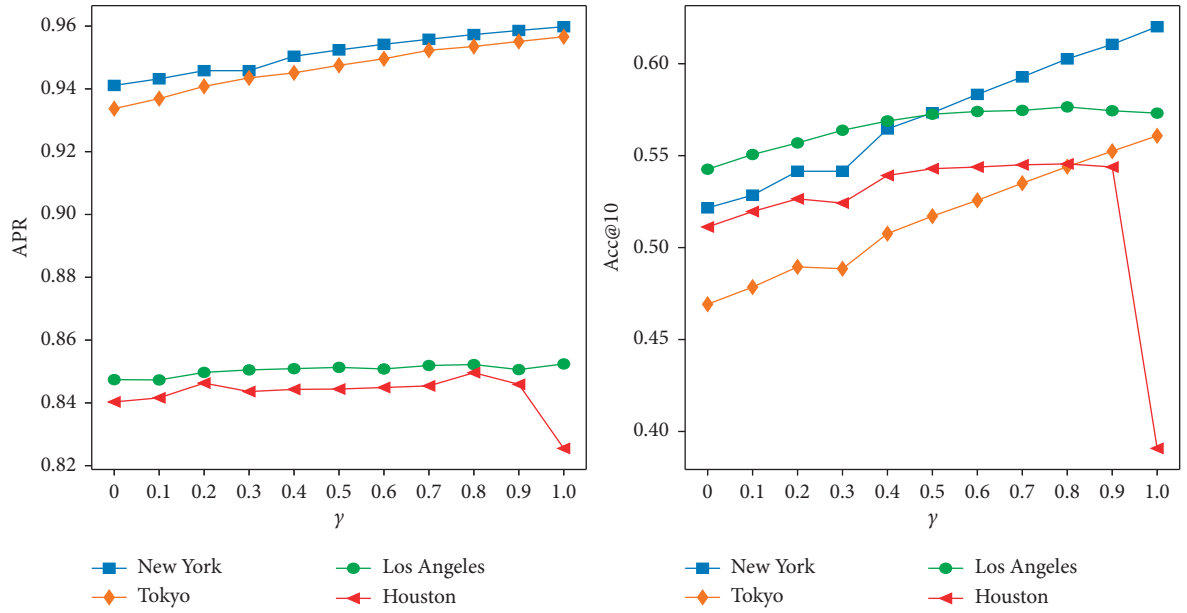


Figure 5: Impacts of spatial and temporal attention.

unchanged. As the factor γ grows, our model performance grows on the Foursquare datasets. However, it can be seen that, in the Gowalla datasets, the upper limit of model performance appears with the increase in factor γ, and when γ takes 1.0 in the Houston dataset, there is a significant decline, which shows that γ has an optimal value 0.9. The reason for this phenomenon may be that the Gowalla datasets are more sparse and the time span is larger than that

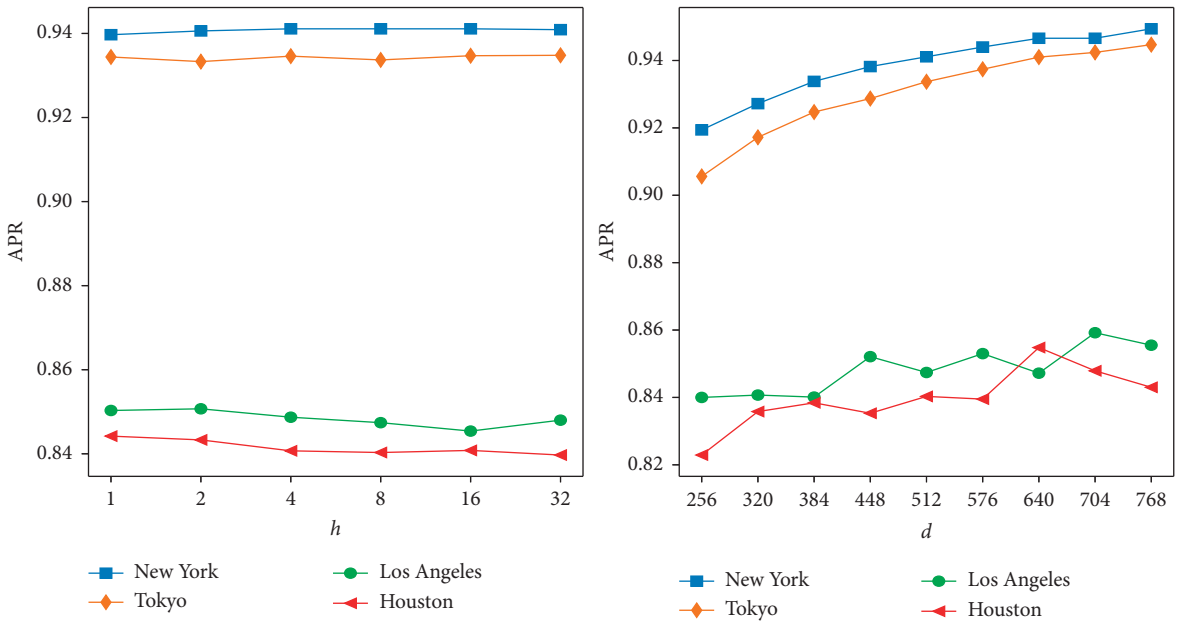


(a)



(b)

Figure 6: Continued.



(c)

Figure 6: Effect of hyperparameter settings: (a) effect of ε and β; (b) effect of γ; (c) effect of h and d.

of Foursquare, which makes the mining of time information more difficult.

5.7.3. Effect of h and d. Figure 6(c) shows the APR performance on four datasets when we vary parameters h and d, respectively. The projection of multiple subspaces can improve the performance, but not obvious, and the optimal h is different in different datasets. On the Foursquare dataset, with the increase in the hidden layer dimension, the accuracy of the model shows an upward trend. However, on the Gowalla dataset, the larger dimension of the model does not bring better accuracy. We analyze that the reason for this may be that more parameters are added to the dimension, which makes the model appear as overfitting problem earlier on the more sparse datasets.

6. Conclusions

In this paper, we proposed a novel network named STSAN (spatial-temporal self-attention network), which can integrate spatial and temporal information with the self-attention for location prediction. Our model can learn the dynamic representation of the user's trajectory by capturing the sequential transitions pattern of the user's trajectory and integrating the user's geographical preference and time correlation. It makes better use of spatial and temporal information to mine the user's trajectory, which is helpful to improve the accuracy of location prediction on sparse data and alleviates the problem of data sparsity. We experimentally evaluate our STSAN model using four datasets collected from Foursquare and Gowalla. The experiment results show the superiority of our approach over various baseline approaches, and it is more prominent in sparse datasets. In addition, we verified the influence of various parameters in STSAN on experimental performance through a large number of experimental results. STSAN introduces the self-attention mechanism and integrates spatiotemporal information, but it introduces a lot of relevant parameters to increase the difficulty in parameter adjustment. In the future, we plan to explore the application of federated machine learning on location prediction and extend our STSAN model to a privacy-aware version. In addition, more contextual information such as knowledge graph is also worth being incorporated into our model in order to enhance the interpretability of the model.

Data Availability

Foursquare dataset contains check-ins in NYC and Tokyo collected for about 10 months (from 12 April 2012 to 16 February 2013). It contains 227,428 check-ins in New York City and 573,703 check-ins in Tokyo. Each check-in is associated with its timestamp, its GPS coordinates, and (https://www.kaggle.com/chetanism/foursquare-nyc-and-tokyo-checkin-dataset) Gowalla dataset. This dataset is collected for about 18 months (from 1 February 2009 to 31 October 2010). It contains 6,442,890 check-ins with its timestamp and its GPS coordinates. We select the check-ins in Los Angeles and Houston as experiment dataset (http://snap.stanford.edu/data/loc-gowalla.html).

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This research was partially funded by the Natural Science Foundation of Liaoning Province (Grant no. 364 2019-MS-

111) and the National Natural Science Foundation of China (Grant no. 61872069).

References

[1] A. Asahara, K. Maruyama, A. Sato, and K. Seto, "Pedestrian-movement prediction based on mixed markov-chain model," in *Proceedings of the 19th ACM SIGSPATIAL International Symposium on Advances in Geographic Information Systems, ACM-GIS 2011*, pp. 25–33, ACM, Chicago, IL, USA, November 2011.

[2] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized Markov chains for next-basket recommendation," in *Proceedings of the 19th International Conference on World Wide Web, WWW 2010*, pp. 811–820, Raleigh, NC, USA, April 2010.

[3] Q. Liu, S. Wu, L. Wang, and T. Tan, "Predicting the next location: a recurrent model with spatial and temporal contexts," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pp. 194–200, AAAI Press, Phoenix, AZ, USA, February 2016.

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

[5] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, https://arxiv.org/abs/1412.3555.

[6] J. Feng, Y. Li, C. Zhang et al., "Deepmove: predicting human mobility with attentional recurrent networks," in *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018*, pp. 1459–1468, Lyon, France, April 2018.

[7] Z. Chang, J. Bai, J. Song et al., "An attention-based user behavior modeling framework for recommendation," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18)*, pp. 4564–4571, AAAI Press, New Orleans, LO, USA, February 2018.

[8] D. Kong, F. Wu, and HST-LSTM, "A hierarchical spatial-temporal long-short term memory network for location prediction," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pp. 2341–2347, Stockholm, Sweden, July 2018.

[9] Di Yao, C. Zhang, J.-H. Huang, and S. E. R. M. Jingping Bi., "A recurrent model for next location prediction in semantic trajectories," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017*, pp. 2411–2414, ACM, Singapore, November 2017.

[10] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Proceedings of the Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pp. 5998–6008, Long Beach, CA, USA, December 2017.

[11] Q. Gao, Z. Fan, G. Trajcevski, K. Zhang, T. Zhong, and F. Zhang, "Predicting human mobility via variational attention," in *Proceedings of the The World Wide Web Conference, WWW 2019*, pp. 2750–2756, ACM, San Francisco, CA, USA, May 2019.

[12] S. Feng, X. Li, Y. Zeng, C. Gao, Y. Meng Chee, and Y. Quan, "Personalized ranking metric embedding for next new POI recommendation," in *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015*, pp. 2069–2075, AAAI Press, Buenos Aires, Argentina, July 2015.

[13] A. Al-Molegi, M. Jabreel, B. Ghaleb, and STF-RNN, "Space time features-based recurrent neural network for predicting people next location," in *Proceedings of the 2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016*, pp. 1–7, IEEE, Athens, Greece, December 2016.

[14] A. Li, S. Wang, W. Li, S. Liu, and S. Zhang, "Predicting human mobility with federated learning," in *Proceedings of the SIGSPATIAL '20: 28th International Conference on Advances in Geographic Information Systems*, pp. 441–444, Seattle, WA, USA, November 2020.

[15] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu, "Recurrent models of visual attention," *Proceedings of the Advances in Neural Information Processing Systems*, vol. 27, pp. 2204–2212, 2014.

[16] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, https://arxiv.org/abs/1409.0473.

[17] M.-T. Luong, H. Pham, and D. Christopher, "Effective approaches to attention-based neural machine translation," 2015, https://arxiv.org/abs/1508.04025.

[18] W. Yin, H. Schütze, B. Xiang, and B. Zhou, "ABCNN: attention-based convolutional neural network for modeling sentence pairs," 2015, https://arxiv.org/abs/1512.05193.

[19] P. Zhou, W. Shi, J. Tian et al., "Attention-based bidirectional long short-term memory networks for relation classification," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Berlin, Germany, August 2016.

[20] S. Zhang, Yi Tay, L. Yao, and A. Sun, "Dynamic intention-aware recommendation with self-attention," 2018, https://arxiv.org/abs/1808.06414v1.

[21] W.-C. Kang and J. J. McAuley, "Self-attentive sequential recommendation," in *Proceedings of the IEEE International Conference on Data Mining, ICDM 2018*, pp. 197–206, IEEE Computer Society, Singapore, November 2018.

[22] M. Xia, Y. Cui, Y. Zhang, Y. Xu, J. Liu, and Y. Xu, "Dau-net: a novel water areas segmentation structure for remote sensing image," *International Journal of Remote Sensing*, vol. 42, no. 7, pp. 2594–2621, 2021.

[23] M. Xia, T. Wang, Y. Zhang, J. Liu, and Y. Xu, "Cloud/shadow segmentation based on global attention feature fusion residual network for remote sensing imagery," *International Journal of Remote Sensing*, vol. 42, no. 6, pp. 2022–2045, 2021.

[24] B. Chen, M. Xia, and J. Huang, "Mfanet: a multi-level feature aggregation network for semantic segmentation of land cover," *Remote Sensing*, vol. 13, no. 4, p. 2021.

[25] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013*, pp. 3111–3119, Lake Tahoe, NA, USA, December 2013.

[26] J. Cao, S. Xu, X. Zhu, R. Lv, and Bo Liu, "Efficient fine-grained location prediction based on user mobility pattern in lbsns," in *Proceedings of the Fifth International Conference on Advanced Cloud and Big Data, CBD*, pp. 238–243, IEEE Computer Society, Shanghai, China, August 2017.

[27] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu, "Modeling user activity preference by leveraging user spatial temporal characteristics in LBSNS," *IEEE Transactions on Systems, Man, and Cybernetics Systems*, vol. 45, no. 1, pp. 129–142, 2015.

[28] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: user movement in location-based social networks," in *Proceedings of the 17th ACM SIGKDD International*

Conference on Knowledge Discovery and Data Mining, pp. 1082–1090, ACM, San Diego, CA, USA, August 2011.

[29] H. Kwak, C. Lee, H. Park, B. Sue, and Moon, "What is twitter, a social network or a news media?" in *Proceedings of the 19th International Conference on World Wide Web, WWW 2010*, pp. 591–600, Raleigh, North Carolina, USA, April 2010.

[30] D. Yang, B. Fankhauser, P. Rosso, and P. Cudré-Mauroux, "Location prediction over sparse user mobility traces using rnns: Flashback in hidden states! in Christian Bessiere," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI*, pp. 2184–2190, New York, NY, USA, July 2020.

[31] C. Zhang, K. Zhang, Y. Quan, L. Zhang, Tim Hanratty, and J. Han, "Gmove: group-level mobility modeling using geo-tagged social media," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1305–1314, ACM, San Francisco, CA, USA, August 2016.

[32] A. Noulas, S. Scellato, L. Neal, and C. Mascolo, "Mining user mobility features for next place prediction in location-based services," in *Proceedings of the 12th IEEE International Conference on Data Mining, ICDM 2012*, pp. 1038–1043, IEEE Computer Society, Brussels, Belgium, December 2012.

[33] G. Hinton, N. Srivastava, and S. Kevin, "Neural networks for machine learning lecture 6a overview of mini-batch gradient descent," *Overview of Mini-Batch Gradient Descent*, vol. 14, no. 8, 2012.

[34] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," 2012, https://arxiv.org/abs/1207.0580.

Hindawi
Complexity
Volume 2021, Article ID 6457246, 16 pages
https://doi.org/10.1155/2021/6457246

Research Article

Learning Air Traffic as Images: A Deep Convolutional Neural Network for Airspace Operation Complexity Evaluation

Hua Xie,[1] Minghua Zhang,[1] Jiaming Ge,[2] Xinfang Dong,[1] and Haiyan Chen[2]

[1]*College of Civil Aviation, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China*
[2]*College of Computer Science and Technology/College of Artificial Intelligence, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China*

Correspondence should be addressed to Minghua Zhang; mh.zhang@nuaa.edu.cn

Received 31 July 2020; Revised 17 December 2020; Accepted 9 January 2021; Published 30 January 2021

Academic Editor: Min Xia

A sector is a basic unit of airspace whose operation is managed by air traffic controllers. The operation complexity of a sector plays an important role in air traffic management system, such as airspace reconfiguration, air traffic flow management, and allocation of air traffic controller resources. Therefore, accurate evaluation of the sector operation complexity (SOC) is crucial. Considering there are numerous factors that can influence SOC, researchers have proposed several machine learning methods recently to evaluate SOC by mining the relationship between factors and complexity. However, existing studies rely on hand-crafted factors, which are computationally difficult, specialized background required, and may limit the evaluation performance of the model. To overcome these problems, this paper for the first time proposes an end-to-end SOC learning framework based on deep convolutional neural network (CNN) specifically for free of hand-crafted factors environment. A new data representation, i.e., multichannel traffic scenario image (MTSI), is proposed to represent the overall air traffic scenario. A MTSI is generated by splitting the airspace into a two-dimension grid map and filled with navigation information. Motivated by the applications of deep learning network, the specific CNN model is introduced to automatically extract high-level traffic features from MTSIs and learn the SOC pattern. Thus, the model input is determined by combining multiple image channels composed of air traffic information, which are used to describe the traffic scenario. The model output is SOC levels for the target sector. The experimental results using a real dataset from the Guangzhou airspace sector in China show that our model can effectively extract traffic complexity information from MTSIs and achieve promising performance than traditional machine learning methods. In practice, our work can be flexibly and conveniently applied to SOC evaluation without the additional calculation of hand-crafted factors.

1. Introduction

Airspace is the carrier of air traffic system, and air traffic controllers (ATCos) are responsible for its safe and efficient operation. In order to regulate air traffic safely, airspace is divided into several smaller sectors which are in charge of ATCos. As the air transport industry is developing rapidly, the surging flight volume and limited airspace have imposed a higher workload on ATCos. According to researches, the high workload of ATCos is more likely to lead to operational errors [1]. Therefore, evaluating and monitoring the ATCos workload is an important prerequisite for safe and effective air traffic management. Meanwhile, intending to properly divide airspace sectors and efficiently manage air traffic flow so that the traffic control workload of ATCos can be kept below the maximum limit, it is necessary to determine an authoritative indicator that can reflect sector control workload accurately and objectively [2].

According to previous studies, it has been shown that air traffic complexity (a.k.a. airspace complexity or air traffic control complexity), which is used to measure the difficulty and efforts required in managing air traffic safely and orderly, might play an important role in the sector traffic control workload [3]. For several years, many researchers have been mining the relationship between air traffic complexity and workload [4–6]. The prevailing view is that the workload of ATCos is a subjective factor and is highly dominated by air traffic complexity, which is an objective factor [7]. Although air traffic complexity and workload are not completely equivalent, it is reasonable to evaluate the

workload by air traffic complexity. The reason lies in the subjective factor is so uncertain and complex that it is necessary for us to quantitatively evaluate the workload in an ATCo-independent way [8]. Note that we refer to the concept of "sector operation complexity (SOC)" from [2] to represent the air traffic complexity of a sector. SOC is more specific because it specifies the "sector" area rather than a point, an airway, or other airspace elements, and it can also distinguish studies on traffic pattern complexity from our "operational" complexity study [2].

To sum up, SOC domains the air traffic control workload, which is essential in the air traffic operation, leading to its extraordinary role in air traffic management, e.g., airspace reconfiguration, air traffic flow management, and allocation of ATCo resources. Therefore, accurately evaluating the SOC is a hot topic both in research and practical applications [9–11].

For decades, many researches quantitatively evaluate air traffic complexity by studying the internal mechanism of air traffic complexity and modelling from different perspectives [12–14]. Another part of the research believes that air traffic complexity is formed due to the influence of a large number of relevant factors, so we can describe and characterize air traffic complexity by comprehensively considering these factors and studying their relationship [2, 15–17]. However, owing to the numerous nonlinear factors affecting air traffic complexity and the complex internal pattern relationship of air traffic data, it is extremely difficult to describe complexity accurately through rigorous modelling based on a certain perspective, and there are also difficulties in the construction of a complete set of complexity-related factors. In addition, the existing methods mostly rely on the subjective experience or domain knowledge in complexity related factors selection, which might encounter calculation problems in the implementation of actual air traffic management (ATM) applications or when the airspace sector changes.

Facing these problems, this paper aims to propose a novel end-to-end SOC learning framework that can directly extract effective complexity-related features from air traffic data and learn SOC pattern, which can be independent of subjective hand-crafted features and make more accurate and general SOC evaluation.

Motivated by the excellent performance of the deep learning technique on modelling and extracting complicated nonlinear features, we put forward a deep convolutional neural network- (CNN-) based approach to evaluate SOC in a given airspace sector. First of all, since CNN mainly deals with data based on image type, we abstract the air traffic scenario into multichannel images, which are used as the input of the CNN. Then, CNN could automatically extract SOC-related high-level features under the guidance of complexity labels through the convolution and pooling processing methods of the convolution kernel. In that case, the extracted features are input into the fully connected layer to learn the relationship between extracted features and SOC. Finally, the backpropagation algorithm is utilized to continuously adjust the weight of feature learning and full connection layer, so as to learn the SOC pattern and achieve SOC evaluation. The experiments show that our image-based CNN model can automatically extract the effective features and acquire better performance of SOC evaluation than traditional machine learning methods.

The contributions of the paper can be summarized as follows:

(i) A new data representation, i.e., multichannel air traffic scenario image (MTSI), is proposed to describe air traffic scenario, and each channel is proved to be effective.

(ii) Sector operation complexity features of air traffic scenario are extracted automatically using a CNN with a high SOC evaluation accuracy.

(iii) Several model training techniques, such as rotation data augmentation, category balanced sampling, and label smoothing, are utilized to improve model performance.

(iv) The proposed method implements an end-to-end SOC learning framework based on the deep learning, which can achieve a higher SOC evaluation performance without tedious hand-crafted features.

The rest of the paper is organized as follows. Section 2 shows related work. Section 3 makes a data description and proposes a two-step procedure that includes converting air traffic to images and a CNN model for sector operation complexity evaluation. In Section 4, we introduce the experimental configurations and conduct four groups of experiments. The results are analysed and discussed. Finally, conclusions are drawn with future study direction in Section 5. For the sake of readability, the acronyms used in this paper are summarized in Table 1.

2. Related Work

This section reviews the previous works on air traffic complexity evaluation, which are more general than SOC evaluation and the main development of convolutional neural network.

In the existing literature, there are two main types of research methods that dominate studies in air traffic complexity evaluation: single model-based methods and factor system-based methods. These two groups of related work are categorized in Table 2, so as to sum up their main aspects and highlight their limitations in comparison to the present study. The first method mainly focuses on studying the internal formation mechanism of air traffic complexity, expecting to build a model to quantify the complexity from one specific perspective. For instance, Lee et al. defined the air traffic complexity as the degree of difficulty for ATCos to resolve the potential flight conflicts when new aircraft enters the target airspace, and they proposed an input-output method to evaluate air traffic complexity [12]. Prandini et al. believed that the probability of flight conflicts within a sector can reflect the magnitude of complexity, so they characterized complexity by means of conflict risk estimation [13]. Moreover, the Lyapunov exponent was introduced in the field of air traffic complexity by Delahaye and Puechmorel,

Table 1: List of the acronym used in the paper.

Acronym	Definition
Acc	Accuracy
AdaBoost	Adaptive boosting
ATCOs/ ATM	Air traffic controllers/air traffic management
BPNN	Backpropagation neural network
CK	Cohen's kappa
CNN	Convolutional neural network
DL/ML	Deep/machine learning
FP/FN	False positive/false negative
GNB	Gaussian naïve Bayes
KNN	*K*-nearest neighbour
LLR	Logistic linear regression
MAE	Mean average error
MLP	Multilayer perception
MTSI	Multichannel traffic scenario images
PCA	Principal component analysis
RF	Random forest
RTPE/RTT	Run-time per-epoch/run-time test
SOC/SOCNN	Sector operation complexity/SOC evaluation using CNN
SVM	Support vector machine
TP/TN	True positive/true negative

who proposed the concept of trajectory disorder to measure intrinsic traffic complexity [14, 18]. The above three methods (i.e., conflict resolution difficulty, conflict probability, and Lyapunov exponent) all depicted air traffic complexity from their separate perspectives. However, as air traffic complexity contains large amounts of information and is embedded with sophisticated relationships, it is usually insufficient to evaluate air traffic complexity perfectly by a single indicator or model [19].

For the purpose of overcoming the deficiencies of the first model-based methods from single perspective, an extra category of complexity evaluation approach was put forward by synthesizing multiple complexity-related factors to characterize air traffic complexity. The most famous one is the dynamic density method, which calculated complexity as the sum of various complexity factors with different weights [15], whereas these linear methods cannot precisely evaluate air traffic complexity as these related factors usually interact in a nonlinear way. Subsequently, machine learning methods were adopted as they can handle the nonlinear problem. In 2006, Gianazza proposed to treat the air traffic complexity evaluation as a complexity level classification task and used backpropagation neural network (BPNN) to capture the nonlinear relationship [16]. Later studies inherited the idea of classification problems and attempted to mine more internal pattern complexity from air traffic data. Adaptive boosting learning algorithm [17], semi-supervised learning [20], and transfer learning [2] have been employed and acquired fruitful achievements in the domain of small sample learning area for air traffic complexity evaluation. The above machine learning methods have achieved great results in the air traffic complexity evaluation, but there remain two problems: (1) this type of algorithm is highly dependent on the selection of the hand-crafted feature set, and the quality of the feature set determines the performance of the final complexity evaluation. Nevertheless, it is extremely difficult to identify an intact feature set that fully characterizes air traffic complexity because of the internal pattern complexity of air traffic scenario. (2) Different sectors have different traffic properties and airspace structures, and the characteristics that affect the operation complexity of different sectors will also differ. For example, the complexity of some sectors mainly comes from the maintenance of flight intervals, while other sectors are mainly concentrated on the complexity of traffic conflict avoidance. Different sectors may have inconsistent feature sets, which also lead to uncertainty in the air traffic complexity evaluation. Therefore, the performance of air traffic complexity evaluation for machine learning methods might be limited by the incomplete and uncertain hand-crafted features.

Compared with traditional machine learning methods, deep learning can capture nonlinear and complex feature from high-dimensional data and achieve various successful adoption in applications, such as disease diagnosis and mobile traffic classification [21–23]. Meanwhile, it also has an important characteristic of feature learning; that is, features can be automatically extracted from the original data. Therefore, in the process of model training, we can directly use the features extracted by the deep learning method, without the participation of manual features. In the deep learning field, CNN is an efficient and effective algorithm for image processing and has been widely applied in image classification, object detection, etc. [24–26]. Yuki et al. introduced the deep neural network to automatically extract features from the trajectory images [27]. A recurrent convolutional model for the large-scale visual learning was developed by Donahue et al. [28]. Baccouche et al. proposed sequential 3D-CNN models for human action recognition [29]. In the field of the text classification task, Lai et al. applied a recurrent architecture to capture contextual information [30]. The results demonstrate that the convolutional neural network is suitable for different types of complex scenes and can automatically learn features from raw data with better performance.

In summary, abstracting the air traffic complexity evaluation problem as a complexity level classification problem achieved considerable results by machine learning methods, but it is faced with the fact that the performance is dependent on hand-crafted features and the existing feature set is subjective, uncertain, and not necessarily complete. The deep learning method has an excellent ability to mine internal pattern complexity and can automatically extract features from raw data. Therefore, we apply the deep learning technique to the problem of air traffic complexity evaluation, which can free from the limitations of hand-crafted features.

3. Materials and Methods

Since the existing SOC-related factors might be not comprehensive, we have to explore other ways to sufficiently mine more knowledge for better SOC evaluation performance. SOC is originated from ATCos, and they manage air

Table 2: Summary of previous works in air traffic complexity evaluation.

Category	Related research	Paper
Single model	Difficulty to resolve potential flight conflicts	Lee et al. [12]
	Probability of flight conflicts	Prandini et al. [13]
	Lyapunov exponent of trajectory	Delahaye and Puechmorel [14, 18]
Factor system	Dynamic density method (linear)	Kopardekar and Magyarits [15]
	PCA + BPNN	Gianazza [16]
	Genetic algorithm + AdaBoost	Xiao et al. [17]
	Semisupervised and active learning	Zhu et al. [20]
	Transfer learning	Cao et al. [2]
Image-based factors	Deep convolutional neural network	This paper

traffic operation based on the radar screen, which displays air traffic situation in the form of video, i.e., continuous multiframes of images. Therefore, we could convert the air traffic scenario information into images and then use deep learning technique to extract useful information. Considering the image-based method, we propose an end-to-end learning framework for evaluating sector operation complexity (SOC) by using the deep convolutional neural network (CNN) learning strategy and name the framework as SOCNN (SOC + CNN). Figure 1 demonstrates the whole scheme of the proposed SOCNN, which is composed of three procedures ((1) data preprocess; (2) MTSI generation; (3) CNN training). It is noted that, in this paper, we will use MSTIs as model input to replace traditional hand-crafted features. As a result, CNN can automatically extract knowledge from MSTIs to achieve the feature learning process and use the learned features for SOC evaluation, which is the novel feature of our proposed SOCNN.

3.1. Data Description. Air traffic data are mainly divided into static airspace data and dynamic flight data. The static data are composed of latitude and longitude data, which is used to separate the airspace structure and set air routes and positioning points. The dynamic data are obtained through radar equipment or ADS-B transmission equipment, covering the main air traffic information. The dynamic data own a wealth of aircraft operation information, including aircraft identification number, latitude, longitude, speed, altitude, and heading. The dynamic flight data used in this paper come from radar, which are collected every 4-5 seconds, including flight position information and flight status information.

In the traditional process of evaluating complexity based on machine learning methods, static data are used to filter the dynamic flight data in the target sector, and then, the filtered dynamic flight data are used to calculate complexity-related features, so as to realize the complexity evaluation. For our MTSI-based deep learning method, static data are mainly used for gridding the airspace, and then, dynamic flight data are filled into the gridded airspace using a certain method to generate traffic scenario images and then use the generated images to perform feature extraction to complete the complexity evaluation task.

The complexity label used in our experiment is obtained through field collection. We invited several controllers of similar experience and age as air traffic control experts to evaluate the complexity of different traffic scenarios. The complexity range in this paper is set as five levels, and the traffic scenario of one-minute time period is an evaluation sample. In order to avoid cognitive differences between different people, we have adopted the method of collecting multiple sets of labels on the same sample to reduce human error.

3.2. Converting Air Traffic to Images. As previously described, SOC is uncertain and changing over time. It could be affected by other factors besides the number of aircraft in the sector, such as the aircraft motion parameters, the relative trends between different aircraft, and the sector entry point of aircraft. Meanwhile, we should not only use the local status information of a single aircraft but also look at the future development of the traffic situation from a global perspective. Therefore, we propose a new data representation called multi-channel air traffic scenario image (MTSI) to represent the overall air traffic scenario and the interactive influence among aircraft in a sector.

Image is formed by a two-dimensional matrix, so we need to grid the target sector first as the basis for subsequent images. In order to ensure the regularity of the image and the convenience of subsequent image operations, we use a circumscribed square of the sector boundary as the range of the image and divide the target sector into grid maps with a suitable scale. The time span of our single air traffic scene sample is 1 minute. Considering that the actual radar data are updated every 4-5 seconds and the average flight speed of aircraft is 15 km per minute, there is only one flight trajectory data every 1–1.25 km. To ensure the existence of real traffic data at every grid, in other words to prevent the phenomenon of crossing the grid, the appropriate grid width should be set within the range of 1.25–15 km. Based on the above factors, in order to show the flight trajectory of the aircraft as carefully as possible and the convenience of calculation, we set the width of the grid at 2 km. There is a spatial position relationship between different grids, so we can map the position of aircraft in the airspace to the corresponding grid position, in which the corresponding grid is filled with the flight status information of the aircraft, such as speed, altitude, and heading. However, since a grid can only be filled with one value and cannot contain a large amount of information at the same time, we use multiple

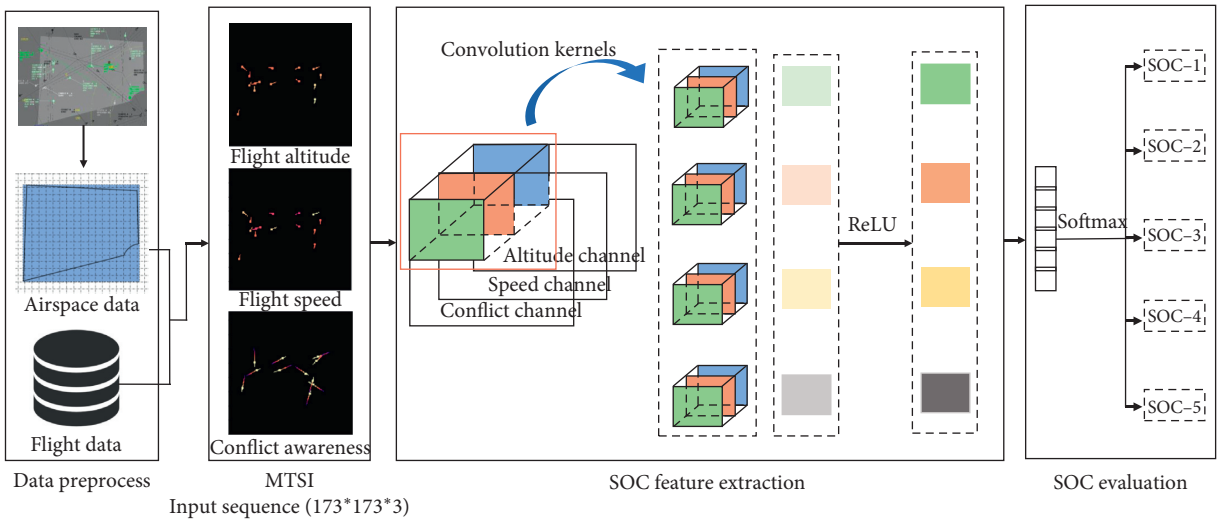


Figure 1: Architecture of the proposed SOCNN.

two-dimensional matrices to store the input of traffic information, respectively. These different two-dimensional matrices can be understood as multichannel images, which we call as multichannel air traffic scenario image (MTSI). These different channels of MTSI express the traffic information of the same traffic scene from different perspectives. When these channels are combined and superimposed at the same time, the real traffic scenario can be restored.

As the traffic scenario in our problem is a period of time rather than a moment, traffic complexity is also not a short-term and instantaneous indicator. In order to reflect the real situation of the aircraft in the period of time, we choose to map all the traffic data received during this period to the image one by one. In other words, through mapping of fight status data in the sector to the corresponding position of the two-dimensional grid matrix, the image will show the historical trajectory of different aircraft, and the grid of the corresponding trajectory is filled with different flight status information. Here, we choose to utilize the speed and altitude traffic information to generate two kinds of images, which are called altitude channel and speed channel, shown in Figure 2.

Furthermore, in order to reflect the operational situation and flight conflict information of the air traffic in a sector, we also construct an image of the unreal trajectory (the predicted trajectory). The predicted trajectory of aircraft is generated by using speed, heading, latitude and longitude information, and simultaneously mapped to the flight conflict awareness channel. Since the predicted trajectory is not completely accurate by the affection by other factors, we think that the influence of the predicted trajectory would become smaller with the increase in the predicted time. To distinguish the magnitude of the influence of the predicted trajectory at different time lengths, we have performed a weakening treatment on the predicted trajectory. Specifically, the start point grid of the predicted trajectory is filled with the maximum pixel value, and then, the grid is filled in the direction of the predicted trajectory. Whenever a new grid is filled, the pixel value will be reduced to a certain extent, until the pixel value is reduced to zero or the predicted duration limit is reached, so as to achieve the weakening effect of the predicted trajectory (see Figure 3(a)).

Based on the actual situation of actual air traffic control, the predicted trajectory time is set to 3 minutes and a predicted trajectory will occupy approximately 20–30 grids according to the grid width setting in the previous part. In order to reflect the gradual weakening of the influence of the predicted trajectory, the starting trajectory point should be set with an appropriate initial value and then gradually decreased. At the end of the predicted trajectory, the grid value should be close to 0. We set the initial value to 10000 and the decline rate to 100. With the passage of the predicted trajectory, the decline rate is dynamic, that is, every decline, and the decline rate increases by 40, so as to reflect the actual situation of the rapid weakening of the real trajectory influence. According to the above settings, the last predicted trajectory grid value will approach 0.

In addition, as aircraft intersection conflict might be an important factor affecting SOC, in order to describe the information, we have carried out pixel enhancement processing on the grid of intersection conflict points of the predicted trajectory (see Figure 3(b)). The steps are as follows: (1) locating of the predicted trajectory conflict grid firstly, (2) extracting the altitude information of the intersection conflict grid of the corresponding aircraft from altitude channel, and (3) determining the grid pixel enhancement value of the intersection conflict point according to

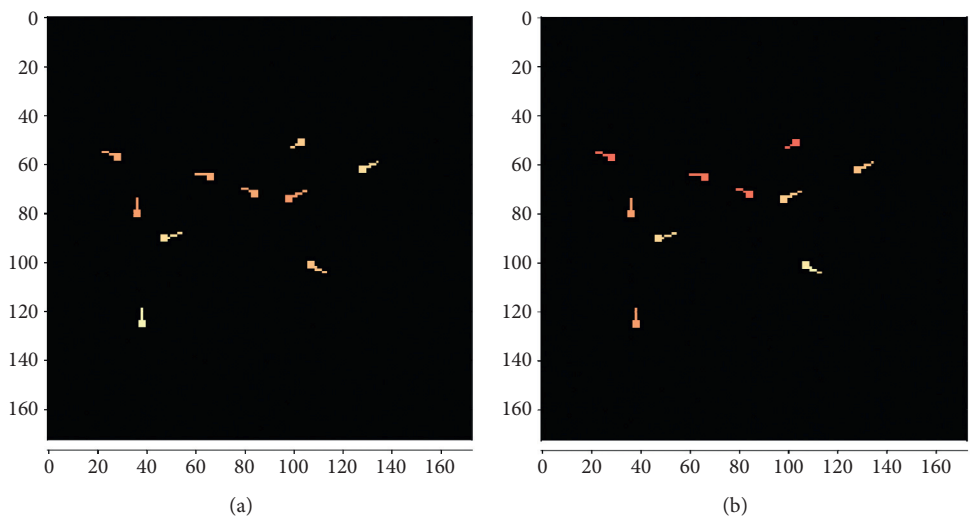


Figure 2: Channels of historical trajectory: (a) altitude channel; (b) speed channel.

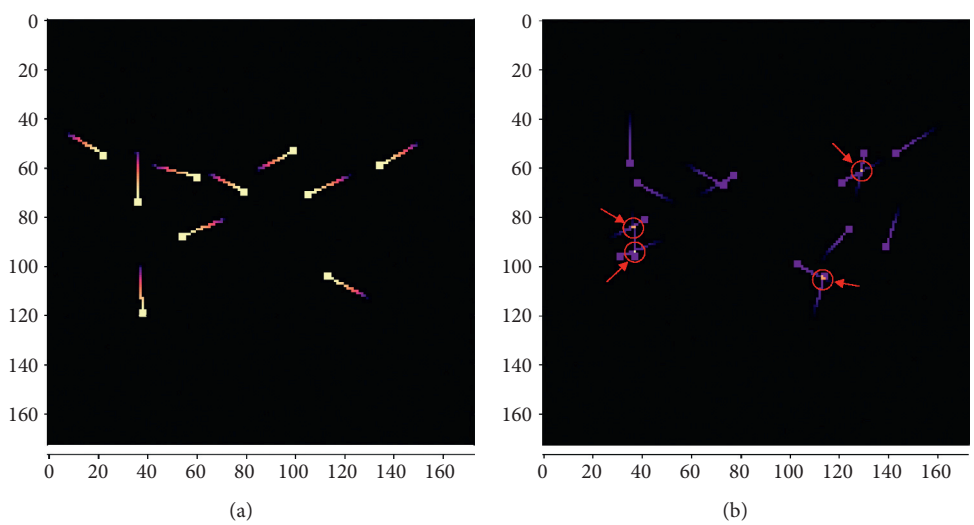


Figure 3: Channel of predicted trajectory: (a) no conflict scenario; (b) conflict scenario.

$$\text{enhancement pixel} = \begin{cases} \left(\text{intersection pixel}_i + \text{intersection pixel}_j\right) * 2, \ \text{altitude difference}_{ij} \leq 300\,m, \\ \left(\text{intersection pixel}_i + \text{intersection pixel}_j\right) * 1.8, \ 300\,m < \text{altitude difference}_{ij} \leq 900\,m, \\ \left(\text{intersection pixel}_i + \text{intersection pixel}_j\right) * 1.6, \ 900\,m < \text{altitude difference}_{ij} \leq 1500\,m, \\ \left(\text{intersection pixel}_i + \text{intersection pixel}_j\right) * 1.4, \ 1500\,m < \text{altitude difference}_{ij} \leq 2100\,m, \\ \left(\text{intersection pixel}_i + \text{intersection pixel}_j\right) * 1.2, \ \text{altitude difference}_{ij} > 2100\,m, \end{cases} \tag{1}$$

where intersection pixel$_i$ denotes the pixel value of the intersection of ith aircraft in predicted trajectory channel and altitude difference$_{ij}$ is the altitude difference between ith and jth aircraft in intersection grid.

Consequently, several single channels are encoded as a multichannel image whose pixel values are represented by various air traffic data. It should be noted that three channel images are used in our final model, namely, altitude, speed, and flight conflict awareness channel. However, more channels can be constructed to accommodate comprehensive air traffic information in future study.

3.3. CNN for Sector Operation Complexity Evaluation

3.3.1. Basic Principle of CNN. Convolutional neural network (CNN), which is proposed to extract high-level features of spatial dependencies, is mainly used in the field of image

recognition. It refers to a kind of network, rather than a certain network, which contains many different structures. Different network structures often behave differently. A typical CNN consists of three parts: the convolutional layer, the pooling layer, and the full connection layer. The convolutional layer is responsible for mining spatial correlations between adjacent grids and extracting local features in the MTSI. The pooling layer is used to mine discriminative features and significantly reduce parameter magnitude. The full connection layer is the part of a traditional neural network that outputs the desired result.

In order to extract diverse features for modelling spatial correlations, a large number of different convolution kernels would be designed to work together. The weights among interdependent adjacent grids are shared so that information learned from one local area can be applied to other parts of the image, which makes the feed-forward propagation and backward training more efficient [31]. The above weight sharing and local perception enable CNN to learn more basic features at the shallow level and maintain the rotation, distortion, and scaling invariance for spatial modelling:

$$x_j^l = f\left(\sum_{i=1}^{N_{t-1}}\left(x_i^{l-1} * k_{ij}^l\right) + b_j^l\right), \tag{2}$$

$$x_{ij}^l = g\left(x_\varsigma^{l-1}\right). \tag{3}$$

Equations (2) and (3), which represent the operation of the convolution layer and pooling layer, respectively, constitute the operation of the feedforward. x_i^l represents the jth feature map of the lth layer. The number of feature maps on the upper layer is denoted by N_{t-1}. The convolutional kernel k_{ij}^l used for image feature extraction and its corresponding offset term is represented by b_j^l. Operator $*$ denotes the convolution operation. $f(\cdot)$ is the nonlinear activation function, such as sigmoid, tanh, and ReLU (rectified linear unit). After completing the convolution operation, it enters the sampling process. The sampling layer realizes the downsampling of all input feature maps, so as to meet the requirement of the invariant feature scale. Unlike the convolutional layer operation, the downsampling layer does not change the number of feature maps but only its size, as shown in equation (3). ς indicates the spatial field of the pooling operation. $g(\cdot)$ is the downsampling function in a pooling layer, which is usually specified as average, median, or maximal operation. When the whole network structure model completes the convolution and pooling operation, all the feature maps are transformed into an intermediate value transition and finally expanded into a one-dimensional vector, which is used as the input of the next neural network, and finally, the classification results are obtained.

3.3.2. SOCNN Network Architecture and Model Training. As described before, in order to meet the CNN input data format requirements, we convert the air traffic data into MTSI, which can include the most basic navigation elements (i.e., altitude, speed, and heading) and conflict awareness capabilities. Considering that the deep learning method is first introduced into SOC evaluation, we design a concise deep convolutional neural network, which is shown in Figure 4. It is used to learn and predict the operation complexity of a certain airspace sector under the premise of given navigation information. The input data are multi-channel input images (MTSIs) converted from air traffic data, and the transformation process can be referred to Section 3.2. The output label data are sector operation complexity level provided by ATCos based on real-time air traffic scenario judgement.

The model consists of several convolutional layers, pooling layers, and full-connected layers. Drawing on the idea of VGG [32], we adopt small convolution kernels ($3 * 3$) and use a number of successive convolutional layers to replace large convolution kernel because the multilayer nonlinear layers can increase the depth of the network to ensure the learning of more complex patterns, and the computation cost is also lower. The number of convolution kernels in each convolutional layer is (32, 32, 64, 64, 128, 128), and the maximum pooling size is ($2 * 2$). The convolution process uses the "SAME" mode. The learning rate is 0.001, and the batch size is 50. The activation function of ReLU is adopted. The goal of deep learning model training is to iteratively optimize the parameters of the network model and learn the distribution of data from the training set samples. In general, the optimization direction is determined by the objective function, which consists of an error item (J) and a regularization item (R):

$$\theta = \arg\min_\theta L(X, Y) = \arg\min_\theta (J + \lambda R), \tag{4}$$

$$J = -\frac{1}{m}\sum_{i=1}^{N} f\left(y(i), y\prime(i)\right). \tag{5}$$

In equation (4), X and Y are the input and output of the model. The loss function and regularization item are represented by J and R, respectively. θ denotes the parameters of a deep neural network, and λ determines the weight of regularization item. The cross entropy is employed as the loss function in this paper and confirmed by $f(\cdot)$, in which $y(i)$ and $y'(i)$ are the ground truth label and predicted output of the ith training sample. The dropout layer is employed to effectively alleviate the occurrence of overfitting. The Adam optimizer is applied to improve the performance of gradient descent algorithm.

The actual SOC dataset has the problems of limited sample size, data imbalance, and label noise. Given the above problems, we adopted several techniques to solve these defects during the model training process. On the problem of limited sample size, data augmentation is an extremely important step in deep learning, and we use the random rotation of images to increase the diversity of our MTSI samples. Considering the problem of data imbalance, we adopt the category equalization sampling technology and equalize the sampling when generating each minibatch to ensure that the learning process will not be biased towards categories with more samples. In order to prevent the model from overlearning noisy samples, we perform label

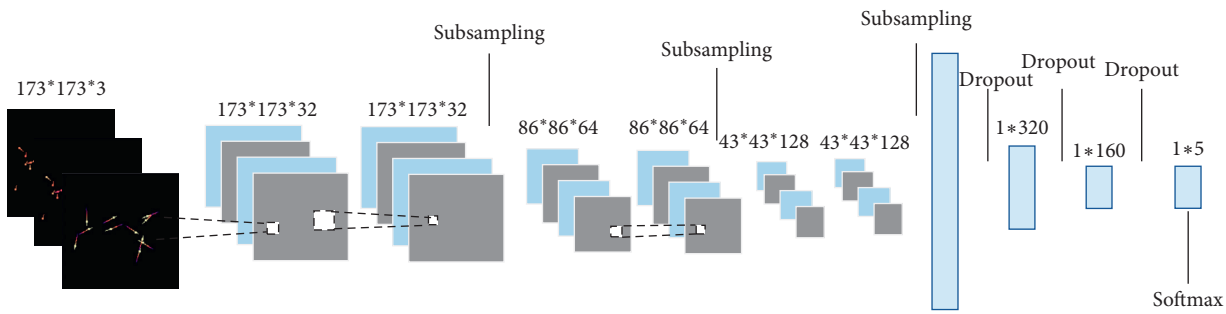


Figure 4: Network architecture of our convolutional neural network.

smoothing processing on the input data, which helps to improve the robustness of its learning process.

The computational complexity of our proposed network architecture can be regarded as the accumulation of the computational complexity of all convolutional layers and represented as $O\left(\sum_{l=1}^{D} M_l^2 \cdot K_l^2 \cdot C_{l-1} \cdot C_l\right)$. Among them, D is the number of layers in the neural network; l represents the l-th convolutional layer of the network; M is the side length of the feature map output by convolution kernel and is mainly determined by input matrix size X, convolution kernel size K, padding, and stride; K is the side length of each convolution kernel; and C_l and C_{l-1}, respectively, represent the number of convolution kernels of the l-th convolutional layer and the number of output channels of the $(l-1)$-th layer.

4. Experiments and Discussion

4.1. Experimental Configurations. In this section, in order to verify the effectiveness of our proposed complexity evaluation method based on deep convolutional neural network, several experiments are conducted on the real air traffic operation data. The target airspace, as shown in the yellow part of Figure 5, is located in the main air route from Guangzhou to Wuhan in China. We collected and filtered out 3605 samples (sample category distribution: SOC-1: 19, SOC-2: 742, SOC-3: 1787, SOC-4: 1010, and SOC-5: 47) of this sector from December 1st to December 15th in 2019. Each sample corresponds to a generated MTSI originated from one-minute air traffic data and a corresponding complexity level (five levels) provided by ATM experts. In the following experiments, the whole dataset is randomly shuffled and divided into two parts. 80% of the samples are training set, and the rest are test set. We also designed several comparison experiments based on machine learning algorithms, in which the hand-crafted features we used have been consistently found to be relevant to air traffic complexity, and their definitions can be referred to [2].

To evaluate the performances of different complexity evaluation models, we employ several criteria, including recall, precision, $F1$-score, accuracy, MAE, and Cohen's kappa (CK). For the criteria definition, the following abbreviations will be used: the number of true positives, TP; the number of false positives, FP; the number of true negatives, TN; and the number of false negatives, FN. Accuracy (Acc) is one of the most commonly used metrics for evaluating the overall performance of classification problems and is the percentage of correctly predicted samples to the total number of samples. Note that the global criterion Acc cannot measure complexity evaluation performance accurately as the category distribution of the sample space is unbalanced. Therefore, the metric of recall, precision, and $F1$ score is introduced. Recall can be thought of percentage of the true samples that are correctly identified by models, and precision focuses on evaluating the proportion of the predicted true samples which are indeed true. The $F1$ score is the harmonic mean of the recall and precision. Cohen's kappa (CK) can also evaluate overall classification performance by consistency. Mean average error (MAE) is metric for regression and is applied to SOC evaluation as there is an ordinal relationship between different complexity levels. The definition of all above metrics is shown as follows:

$$\begin{aligned}
\text{Acc} &= \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}},\\
\text{Recall} &= \frac{\text{TP}}{\text{TP}+\text{FN}},\\
\text{Precision} &= \frac{\text{TP}}{\text{TP}+\text{FP}},\\
F1\text{score} &= 2\times\frac{\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}},\\
\text{CK} &= \frac{\text{Accuracy}-p_e}{1-p_e},\\
p_e &= p_{\text{true}}+p_{\text{false}} = \frac{(\text{TP}+\text{FN})(\text{TP}+\text{FP})}{N^2}\\
&\quad + \frac{(\text{TN}+\text{FN})(\text{TN}+\text{FP})}{N^2},\\
\text{MAE} &= \frac{1}{N}\sum_{i=1}^{N}\left|\hat{y}_i - y_i\right|,
\end{aligned} \tag{6}$$

where $\hat{Y} = \{y_i | i = 1, 2, \ldots, N\}$ denotes the predicted value, $Y = \{y_i | i = 1, 2, \ldots, N\}$ represents the ground truth, and N is the size of test samples. The main configuration of the training server is summarized as follows: 40 $*$ Intel Xeon E5-2640 CPUs, 128 GB memory, 2 $*$ NVIDIA Tesla M60 GPUs, and the operating system is Windows Server 2012 R2.

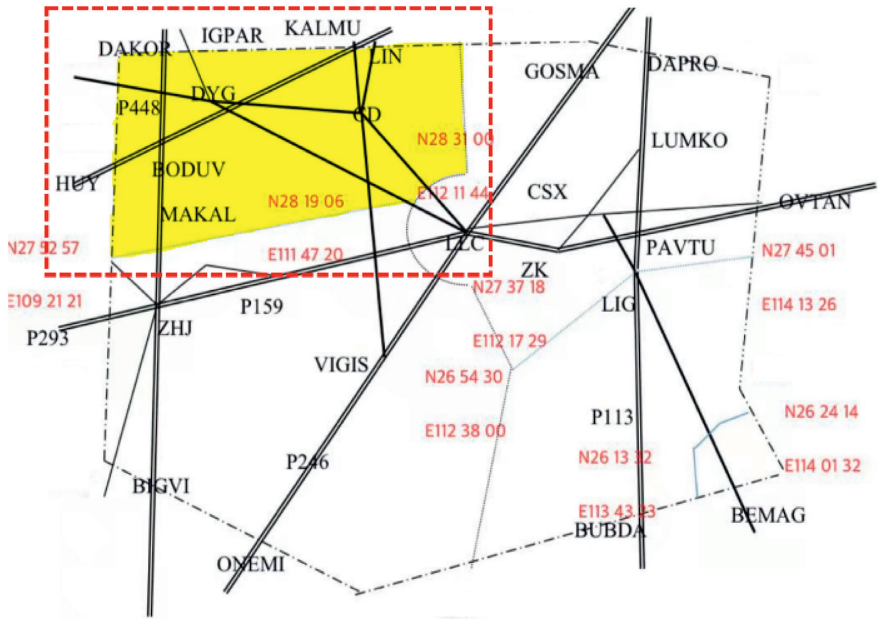


Figure 5: Target airspace sectors structure.

The programming language of all codes is Python, and the deep learning model is built based on the TensorFlow (2.2.0) framework.

4.2. Performance Comparison between Complexity Evaluation Methods. The first experiment focuses on comparing the performance of our SOCNN model, learning from MTSIs, with several machine learning methods based on hand-crafted features. These contrastive machine learning methods include Gaussian naïve Bayes (GNB), k-nearest neighbour (KNN), logistic linear regression (LLR), support vector machine (SVM), multilayer perception (MLP), and ensemble learning algorithms, such as random forest (RF) and adaptive boosting (AdaBoost). Their parameters have been adjusted by the grid search method. In order to measure the generalization capability of the model more rigorously and avoid the particularity brought by the fixed division of small datasets, we conducted a (stratified) five-fold cross-validation and provided the mean and standard variance of each performance measure on the five different folds. Considering the limited space and the conciseness of the result presentation, we have selected the three most important metrics (i.e., accuracy, $F1$ score, and MAE) to study the performances of different methods, which are shown in Table 3.

From the results above, we have the following observation:

(i) Compared to these machine learning methods, SOCNN acquires the best result on the three performance criteria, i.e., Acc, $F1$ score, and MAE. The main difference between SOCNN and machine learning methods lies in the features used. Among them, SOCNN automatically extracts features from MTSIs through a deep convolution neural network, while machine learning methods use hand-crafted features. Even if excellent algorithms such as ensemble learning are used, the performance gap still remains from our SOCNN method. The result demonstrates that the existing hand-crafted features might be insufficient in describing air traffic complexity, and the deep learning method can extract effective information from the constructed MTSIs.

(ii) In addition to SOCNN, we also used the results of single-layer convolution and pooling CNN (shallow CNN) as a comparison. The result of it is the worst except for GNB. This shows that CNN has the ability to learn SOC pattern, but a shallow network cannot learn high-level knowledge well. Therefore, a more complex CNN network should be constructed to learn the knowledge better.

(iii) Among the machine learning algorithms, AdaBoost has achieved great results, which shows that the ensemble learning method is effective. In addition, SVM and MLP perform better than LLR because the air traffic data have nonlinearity and internal pattern complexity, which cannot be learnt by general linear models. Finally, due to the imbalance problem, the performance of models could not be measured only by Acc. For example, the Acc of SVM and MLP are not much different, but the $F1$ score of SVM is significantly worse, indicating that SVM is more inclined to the majority category and performs poorly in minority categories.

To further study the evaluation performance of different methods, we have grouped six performance metrics by training datasets and test datasets and presented them as radar charts (see Figure 6). The outermost and largest circle would be the perfect score on all metrics.

From the radar chart, it is readily apparent to analyse the overfitting phenomenon of models, in which the scores of

Table 3: Evaluation performance of different methods.

Methods		Accuracy	MAE	$F1$ score
ML (features required)	GNB	58.28% (±1.86%)	0.4422 (±0.0228)	47.56% (±2.78%)
	KNN	72.09% (±1.59%)	0.2891 (±0.0260)	62.60% (±4.42%)
	LLR	68.26% (±1.10%)	0.3212 (±0.0118)	46.31% (±2.72%)
	SVM	71.96% (±1.01%)	0.2874 (±0.0113)	52.22% (±4.14%)
	MLP	71.98% (±1.66%)	0.2885 (±0.0163)	59.46% (±8.19%)
	RF	69.49% (±0.93%)	0.3129 (±0.0008)	59.88% (±4.06%)
	AdaBoost	73.84% (±0.91%)	0.2705 (±0.0080)	64.16% (±9.64%)
DL (MTSI)	Shadow-CNN	61.63% (±1.05%)	0.4132 (±0.0129)	51.23% (±5.42%)
	SOCNN	**76.06%** (±1.04%)	**0.2499** (±0.0080)	**70.23%** (±4.61%)

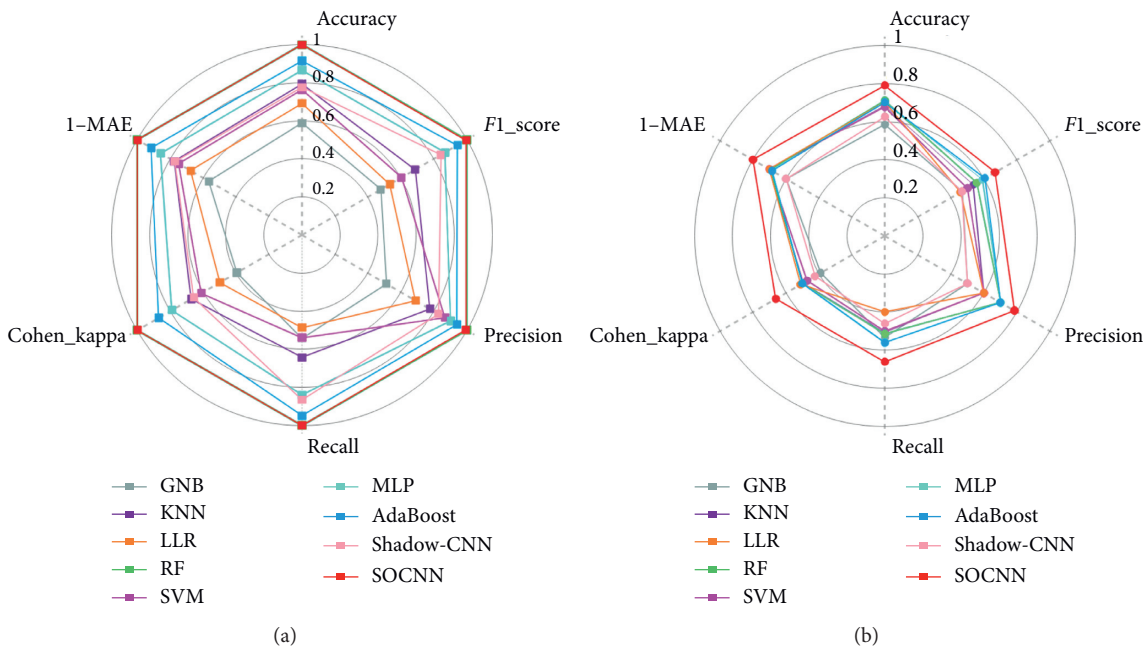


Figure 6: Rader chart of different metrics: (a) training set; (b) test set.

the training set are high, while the test set has low scores. The shape of these irregular polygons can also represent the quality of different algorithms. The larger the polygon area of the test set, the better its performance.

On the radar chart of the training set (see Figure 6(a)), SOCNN, RF, AdaBoost, and MLP all have achieved excellent results. The evaluation metrics such as Acc, recall, precision, and $F1$ score have reached more than 80%, and the accuracy of RF and SOCNN in the training set is almost close to 100%, which reflects that these two algorithms have strong learning capabilities for existing samples. However, the accuracy of the training set does not guarantee that the model has the same performance on unfamiliar samples.

From the radar chart of the test set (see Figure 6(b)), it can be seen that the performance of RF, AdaBoost, and MLP on the test set has dropped significantly. Only SOCNN maintains a relatively high level, in which Acc and precision are all close to 80%. This indicates that the serious overfitting phenomenon has occurred in RF, MLP, and AdaBoost methods. It is also clear that the polygonal figure of SOCNN completely surrounds all the evaluation indicators of other methods, indicating that the performance of SOCNN on the test set is better than all others. AdaBoost performs best in machine learning category methods. GNB and shallow-CNN have the worst performance.

According to the above experimental results, we can see that our method surpasses the traditional machine learning methods in several performance metrics on our dataset. In the same kind of research [2, 19, 20], the complexity evaluation accuracy of existing studies is generally at the level of 70%–80%. It can be seen that the accuracy (76.06%) of our experimental evaluation is comparable to that of existing studies. It should be noted that, in terms of the dataset used, our complexity level is collected at 5 levels, while the existing research has 3 levels, which undoubtedly makes our complexity evaluation task more difficult. Therefore, we believe that our experimental evaluation is meaningful in terms of evaluation performance metrics compared to existing similar studies. In practical application, the air traffic system is a system with a person in the loop. The complexity evaluation results are generally used to provide ATCos with decision-making assistance and reference. In this case, the current

evaluation accuracy is sufficient to meet the needs of practical work, so we think that the experimental evaluation of our method is effective in practice. We are also trying more methods to improve evaluation accuracy to explore other SOC application possibilities in the future.

4.3. Performance Analysis of SOCNN. Figure 7 shows the changes in Acc and loss function on the training set and the test set during the training process of SOCNN. We conducted a 300-epoch experiment. It can be found that, at the 100th epoch, the Acc and loss on the test set have reached the convergence state, and they fluctuate stably in the later stage. The Acc is basically stable between 75% and 80%, while the loss is stable at 1.15–1.20. The situation in the training set is slightly different. The Acc on the training set tends to converge at the 70th–80th epoch, while the loss is still in a declining state at the same time; it gradually reaches the convergent state until the 200th epoch. The above results show that the iterative training process of SOCNN is reasonable and no serious overfitting phenomenon.

Confusion matrix is a performance measurement in classification problems, in which the table has size equal to the number of classes squared. As shown in Figure 8(a), the confusion matrix of SOCNN has high values on the diagonals, and hence, SOCNN is proved to be an efficient method for SOC evaluation. From the previous metric of MAE, it is clear that the average mean error of our SOCNN method is quite small. Here, we further calculated SOC evaluation error distribution based on the confusion matrix (see Figure 8(b)). Different coloured bars represent different degrees of error. In the results of the SOCNN method, 77.1% of the cases are evaluated with the same complexity level as the true complexity level, and 22.2% of the cases have an evaluation error of 1 level. In summary, the evaluation error of 99.3% cases is within 1 level. Only 5 samples, accounting for less than 1%, have an estimated complexity error greater than 1 level and no sample with an error of more than 2 levels. This result once again shows that our SOCNN method not only has a great performance in the overall accuracy but also owns a relatively low prediction error.

4.4. Effectiveness Verification of Multichannel Structure. In this group of experiments, we will verify the effectiveness of our proposed channels and explore the impact of different channel numbers on the performance of SOCNN. First of all, we define the three channels proposed in Section 3.2 as basic channels, in which there is no channel of heading. The reason is we believe that the channel of conflict awareness already contains heading information, but in this experiment, we still take the channel of heading into consideration in order to verify the effects of different channels. So, we currently have a total of 4 channels, namely, channel-altitude (C1), channel-speed (C2), channel-conflict awareness (C3), and channel-heading (C4). According to the number of selected channels, 4 major groups of experiments were designed. The experimental results are shown in Table 4. As a representation of the computational complexity of our model, we report their training time and test time. Specifically, since training is performed on several epochs, we report such information in a normalized way, by providing the run-time per-epoch (RTPE). Similarly, we express the prediction time spent on the test set as the run-time test (RTT).

Through the experimental results obtained, we observe the following:

(i) The single channel group experiment is used to study the utility of single channel for SOC evaluation. This group experiment shows that even when there is only one channel information, the SOC evaluation can reach an accuracy of about 70%. It may be that every single channel is composed of a historical trajectory or predicted trajectory. Although the pixel value of the channel is filled with single navigation information such as altitude or speed, the shape of the trajectory still contains the spatial structure relationship. Convolution and pooling operations of CNN can mine the spatial relationship feature, restore the traffic scenario, and use the extracted features for SOC pattern learning. It is worth noting that C3 channel alone can achieve an accuracy of 72.4%. The reason is that the C3 channel generates a predicted trajectory to provide the ability of conflict awareness. The direction of the predicted trajectory is provided by the heading, and the length is determined by the speed. Therefore, the C3 channel not only contains navigation information such as heading and speed but also has the ability to sense flight conflicts, which make it achieve better performance in SOC evaluation.

(ii) Comparing the experimental results of the two channels group and the single channel group can prove that the combination of two channels is better than the effect of single channel alone, because two channels contain more information and could produce a synergistic joint effect to describe the traffic information together. For example, the joint effect of heading and altitude may determine whether there is a conflict between aircraft.

(iii) Because of the mutual supplement function and joint effect of multichannels, the accuracy is as high as 77.12% when using three channels (C1, C2, and C3), which is the optimal number and combination of channels. However, other experimental results in three channels group experiment are not as expected. From the comparison experiment in this group experiments, we found that due to the addition of the C4 channel, the result is not as well as the previous fewer channels. C4 belongs to the heading channel. The pixel values in the channel are filled with heading data. We originally expected this channel to provide heading information of aircraft for the deep learning process, but the result does not

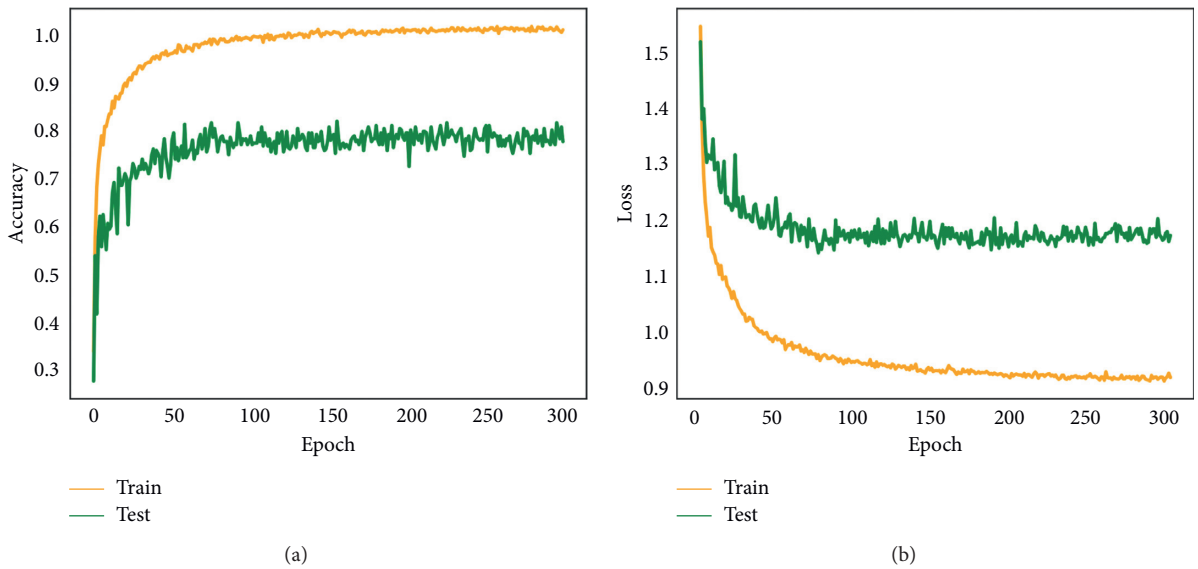


Figure 7: Training process of SOCNN: (a) change in ACC in training; (b) change in loss in training.

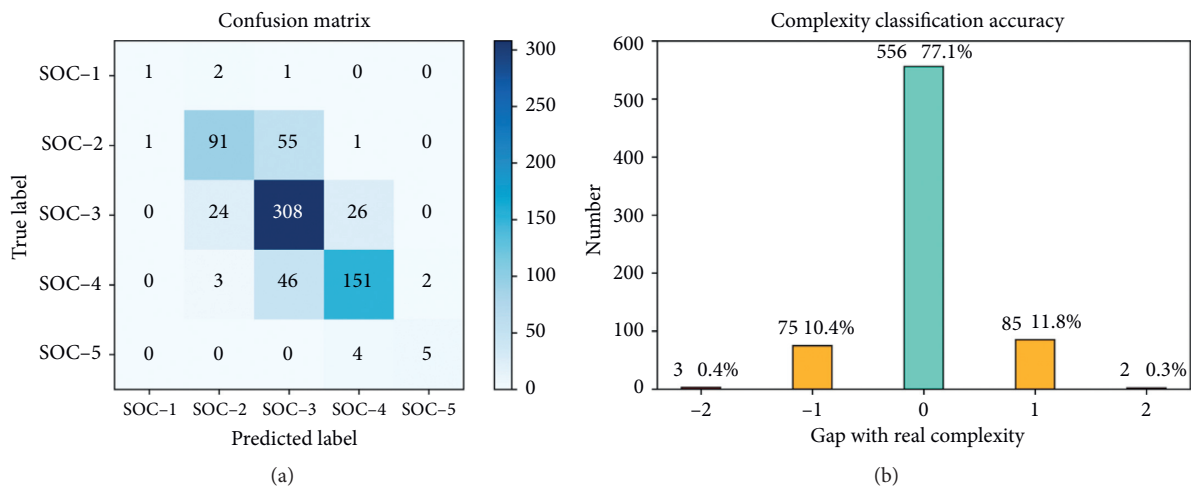


Figure 8: Situation between the predicted SOC and real SOC: (a) confusion matrix of prediction; (b) SOC evaluation error distribution.

Table 4: Performance of different channels combination.

Channels combination		Accuracy (%)	MAE	$F1$ score (%)	RTPE (s)	RTT (s)
Single channel	C1	70.32	0.3093	56.94	19.20	1.19
	C2	69.63	0.3176	44.71	18.79	1.23
	C3	72.40	0.2843	65.83	20.61	1.22
Two channels	C1, C2	70.60	0.3037	50.00	39.58	1.47
	C1, C3	75.59	0.2538	62.46	37.62	1.51
	C2, C3	74.48	0.2635	56.60	37.19	1.45
Three channels	C1, C2, C3	**77.12**	**0.2358**	**69.91**	42.50	1.81
	C1, C2, C4	69.21	0.3190	54.26	44.77	1.86
	C1, C3, C4	74.62	0.2649	61.80	43.68	1.67
	C2, C3, C4	73.09	0.2753	60.37	44.17	1.63
Four channels	C1, C2, C3, C4	74.90	0.2600	65.66	63.79	1.92

seem to be the case. The analysis shows that it is not appropriate to use heading data directly as the pixel values for the channel because the heading data have a special relationship. For example, a heading of 1 degree and a heading of 365 degrees are very similar in actual space, but in terms of the magnitude of the numerical relationship, there is a huge difference between them. It is precisely because of the wrong information provided by heading data that the CNN model might be affected, which reduces the accuracy of the final prediction.

In summary, it can be seen that each channel of C1, C2, and C3 is effective in complexity evaluation, and the combined effect of different channels can improve the evaluation performance of our model, but this does not mean that the more the channels, the better the evaluation performance. The addition of redundant and inappropriate channels, such as C4 channel might harm the evaluation performance of the model.

To investigate the computational complexity of our method, we report their RTPE and RTT of different channel combinations. The result is obviously that the training time and prediction time of the model increase with the increase in the number of channels because the number of input data channels is positively correlated with computational complexity. Taking the C1-C2-C3 channel combination as an example for specific analysis, the average training time of one epoch is 42.50 s and the prediction time on the test set is within 2 s. As can be seen from Figure 7(a), the model generally converges when it is trained to 70–80 epochs, so it will take less than one hour ($42.5\,\mathrm{s} * 80\,\mathrm{epoch} < 1\,\mathrm{h}$) to complete the whole model training process. In the actual air traffic control problem, since the complexity labels are difficult to obtain in real time, historical data are generally used offline to train the model; then, the trained model is used for real-time SOC evaluation. Therefore, the computational cost of the model on the test set is critical in practical application, and our method, within 2 s prediction time, is applicable. If it is necessary to consider the impact of sample updates on the model in the future, high requirements will be put forward for the model training time. Our method can realize the updated sample about one hour before the evaluation time is included in the model training process.

4.5. Research on SOCNN's Parameters. During the construction of the proposed SOCNN, there are several critical parameters that should be properly set up. In this section, we will investigate the range of random rotation angle in data augmentation and label smoothing coefficient in overfitting suppression with respect to their impact on the performance of SOCNN. In these experiments, except for the researched parameters, all of the settings of SOCNN remain the same as in Section 4.2.

4.5.1. Parameter Research on the Range of Rotation Angle. Deep learning requires a large amount of labelled data, but in many cases, the amount of data is insufficient, and our SOC evaluation problem is no exception. Therefore, we adopted a data augmentation strategy to prevent overfitting under conditions of insufficient sample size. Due to the particularity of the SOC evaluation, data augmentation such as random cropping and noise addition is not applicable, and only the random rotation method is used to enhance the diversity of our MTSIs in this paper. In the experiment, we found that the data augmentation of random rotation will indeed improve the performance of SOC evaluation, but the setting of the random rotation angle range will have different effects on the final result. So, we designed a group of experiments to explore how the range of rotation angles influences the performance of SOCNN. The experimental settings are the same as those in Section 4.2 apart from the range of rotation angle and batch size. Here, the range of rotation angle varies from 0 to 360, and experiments of different batch sizes (25, 50, 75, and 100) were conducted to investigate the robustness of our method for each setting. The specific experimental results are shown in Figure 9.

As can be seen from the above figure, when the random rotation angle is set between 0 and 60 degrees, the performance of SOCNN shows a state of rising first and then falling. Judging from the performance metrics (i.e., Acc, MAE, and $F1$ score) we used, it can be considered that, in this interval which we call positive range, the data augmentation operation has indeed improved the overall SOC evaluation performance. When the random rotation range is set to 10, our method has reached the optimal performance. However, when the random rotation angle range exceeds 90 degrees, the overall performance begins to be lower than the case without data augmentation. This phenomenon tells us that, for our SOC evaluation problem, random rotation strategy can certainly impact the performance of the model, and its performance is greatly affected by the setting of the rotation angle range.

Analysing the reasons, it can be seen that as SOC evaluation considers the traffic operation complexity in the whole sector, in order to ensure the overall integrity, the data augmentation methods of zooming, shearing, and panning may not be applicable. The real air traffic operation is based on fixed airways, which is not as strict as ground traffic. The aircraft may not fly completely according to airways, and its flight direction tends to be different from real airway direction, under which our random rotation method can be effective. However, this method is limited. The deviation of airctafts from the airways has to meet the flight requirements. In actual flight, it is almost impossible to cause a large-angle deviation. At the same time, the restriction of airways also ensures that the overall air traffic flow maintains a certain direction. Therefore, the angle of our random rotation cannot be too large. Otherwise, it will produce samples that are completely irrelevant to the actual situation. These samples might misunderstand CNN model and affect the final evaluation performance. From the above experimental results, it can be found that when the random rotation angle is set to 10 degrees, the best performance can be achieved.

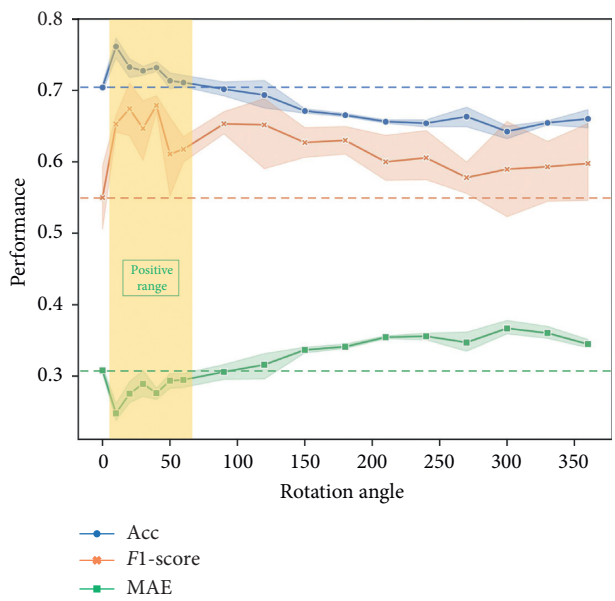


Figure 9: SOCNN's performance change as the range of rotation angle varies.

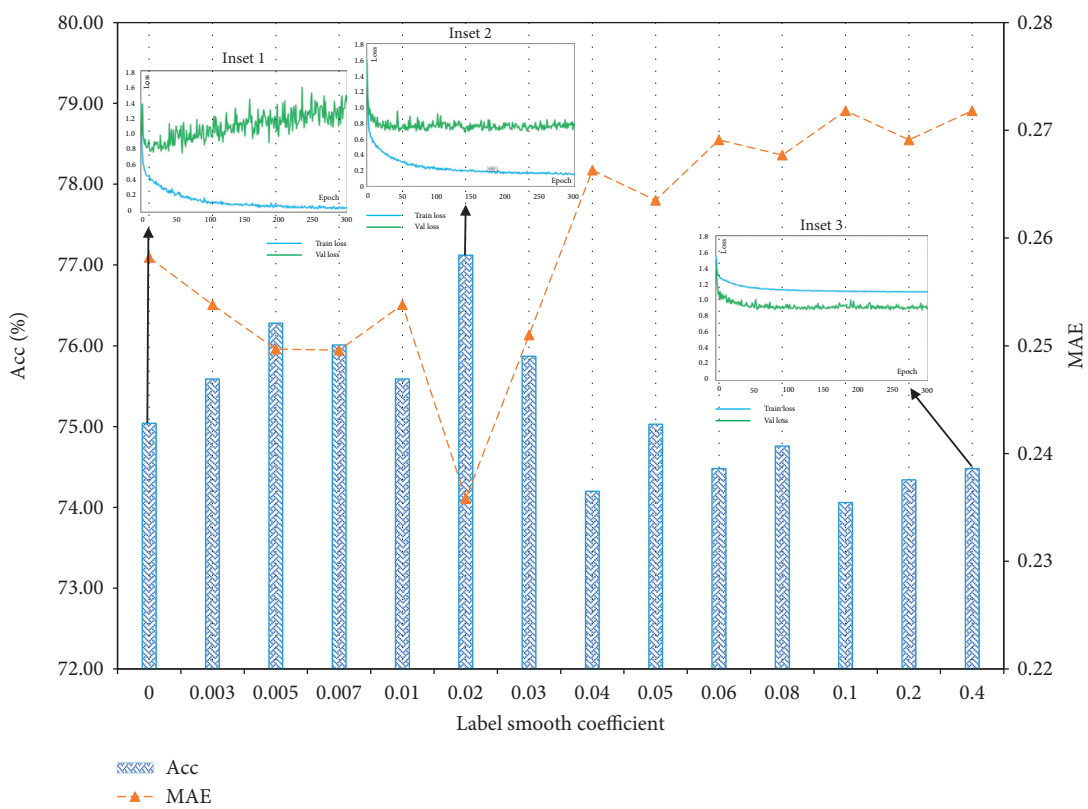


Figure 10: Performance and convergence of SOCNN based on different label smoothing coefficients.

4.5.2. Parameter Research on Label Smoothing Coefficient. The label smoothing strategy is a loss function modification to solve the shortcomings in the process of training deep learning networks, that is, deep neural networks become "overconfident" in their predictions during training, which will reduce their generalization ability. Here, we design a group of experiments to explore the relationships between label smoothing coefficient and the performance of SOCNN. We let the label smoothing coefficient vary from 0 to 0.4 while keeping the other setting unchanged, and then, a total

of 14 experiments were conducted. The metrics of Acc and MAE are utilized for evaluating the performance of SOCNN, and the experimental results are shown in Figure 10.

In Figure 10, the blue histogram and orange dashed line represent the changing trend of Acc and MAE on the test set, respectively. The subgraph denotes the convergence curve of the loss function of the training set and the test set under different parameter settings. If the coefficient is 0, the label smoothing strategy has not been performed. We can find that, in terms of model performance, with the increase in the label smoothing coefficient, the evaluation performance first increases and then decreases. The evaluation performance with a label smoothing coefficient between 0.003 and 0.03 is better than the performance without label smoothing strategy; that is, the coefficient is 0. When the coefficient is greater than 0.03, the model performance will be weakened or even lag behind the performance of the nonlabel smoothing strategy. In terms of the convergence curve of the loss function, it can be seen from the subgraph that the cross-entropy loss curve on the test set cannot converge when the label smoothing strategy is not carried out (as shown in Inset 1 of Figure 10), and the label smoothing strategy is helpful to the loss function convergence of the test set (as shown in Inset 2 of Figure 10), but too large coefficient leads to high loss of training set (as shown in Inset 3 of Figure 10).

The reason for the above phenomenon is that an excessively large label smoothing coefficient will lose part of useful information and reduce the learning ability of the model, thereby affecting the evaluation performance of the model and the abnormality of the loss function convergence curve of the test set. Considering the above analysis results and real experimental results, we choose a label smoothing coefficient of 0.02, which can not only ensure the improvement of model performance but also make the loss curve of the test set converge correctly.

5. Conclusions

Deep learning techniques are widely used in the field of image processing and have achieved fruitful results because of its powerful complex feature representation capabilities than other methods [33, 34]. However, there are limited studies in SOC evaluation. Extracting more complex features by deep learning methods will improve the performance of SOC evaluation.

This paper proposes an image-based SOC evaluation method that can automatically extract abstract traffic features to learn SOC pattern. The method mainly consists of two parts. The first one involves converting air traffic scenario to the multichannels image that contains navigation and conflict information. The second procedure is to utilize a deep CNN to learn airspace operation complexity information based on the constructed multichannels image and realize SOC evaluation. In the experimental results, our methods outperform other prevailing machine learning methods among all performance metrics and every channel of the image is proved to be effective. In addition, we also performed parameters analysis on data augmentation and label smoothing.

Due to the implementation of the end-to-end learning framework, the proposed method can be applied more easily in practice than traditional machine learning methods, which rely on mass hand-crafted features and have difficulty in feature calculation. Moreover, we believe that our method can be further improved in the future in the following directions: (1) we can attempt to design more complex and efficient networks, such as ResNet, DenseNet, and MobileNet, to further improve the SOC evaluation performance and efficiency; (2) the real air traffic scene is not based on a single frame image, and the 2D image may not take into account the motion information between frames in the time dimension, so 3D CNN or Conv-LSTM method can be used to better capture the temporal and spatial feature information in the air traffic scenarios; (3) since it is difficult to obtain labelled samples of the target sector, we can try to build a more accurate SOC evaluation model by making use of unlabelled samples of target sector or labelled sample of nontarget sectors in the case of limited labelled samples through semi-supervised learning or transfer learning techniques.

Data Availability

The data used to support the findings of this study are currently under embargo, while the research findings are commercialized. Requests for data, 12 months after publication of this article, will be considered by the corresponding author.

Conflicts of Interest

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

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. 61903187) and Nanjing University of Aeronautics and Astronautics Graduate Innovation Base (Laboratory) Open Fund (no. kfjj20190732).

References

[1] M. Hansen and Y. Zhang, "Safety of efficiency: link between operational performance and operational errors in the national airspace system," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1888, no. 1, p. 15, 2004.

[2] X. Cao, X. Zhu, Z. Tian, J. Chen, D. Wu, and W. Du, "A knowledge-transfer-based learning framework for airspace operation complexity evaluation," *Transportation Research Part C: Emerging Technologies*, vol. 95, pp. 61–81, 2018.

[3] M. Prandini, L. Piroddi, S. Puechmorel, and S. L. Brazdilova, "Toward air traffic complexity assessment in new generation air traffic management systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 3, pp. 809–818, 2011.

[4] G. Chatterji and B. Sridhar, "Measures for air traffic controller workload prediction," in *Proceedings of the 1st AIAA, Aircraft,*

Technology Integration, and Operations Forum, p. 14, Los Angeles, CA, USA, 2001.

[5] P. Kopardekar and S. Magyarits, "Dynamic density: measuring and predicting sector complexity," in *Proceedings of the 21st Digital Avionics Systems Conference*, vol. 1, p. 2C4, Irvine, CA, USA, October 2002.

[6] A. J. Masalonis, M. B. Callaham, and C. R. Wanke, *Dynamic Density and Complexity Metrics for Realtime Traffic Flow Management*, MITRE Corporation, McLean, VA, USA, 2003.

[7] P. Kopardekar, J. Rhodes, A. Schwartz, S. Magyarits, and B. Willems, "Relationship of maximum manageable air traffic control complexity and sector capacity," in *Proceedings of the 26th International Congress of the Aeronautical Sciences*, Anchorage, AK, USA, 2008.

[8] S. M. A. Rahman, C. Borst, M. Mulder, and R. Paassen, "Sector complexity measures: a comparison," *Jurnal Teknologi*, vol. 76, no. 11, pp. 131–139, 2015.

[9] Z. Song, Y. Chen, Z. Li, D. Zhang, and H. Bi, "Measurement of controller workloads based on air traffic complexity factors," in *Proceedings of the 12th COTA International Conference of Transportation Professionals*, Beijing, China, 2012.

[10] V. F. Gomez Comendador, R. M. Arnaldo Valdés, A. Vidosavljevic, M. Sanchez Cidoncha, and S. Zheng, "Impact of trajectories' uncertainty in existing ATC complexity methodologies and metrics for DAC and FCA SESAR concepts," *Energies*, vol. 12, no. 8, p. 1559, 2019.

[11] A. Majumdar and W. Y. Ochieng, "Air traffic control complexity and safety: framework for sector design based on controller interviews of complexity factors," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2007, no. 1, 2007.

[12] K. Lee, E. Feron, and A. Pritchett, "Describing airspace complexity: airspace response to disturbances," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 1, pp. 210–222, 2009.

[13] M. Prandini, V. Putta, and J. Hu, "A probabilistic measure of air traffic complexity in 3-D airspace," *International Journal of Adaptive Control and Signal Processing*, vol. 24, no. 10, pp. 813–829, 2010.

[14] D. Delahaye and S. Puechmorel, "Air traffic complexity: towards intrinsic metrics," in *Proceedings of the 3th USA/Europe Air Traffic Management R&D Seminar*, Sante Fe, NM, USA, 2000.

[15] P. Kopardekar and S. Magyarits, "Measurement and prediction of dynamic density," in *Proceedings of the 5th USA/Europe Air Traffic Management R&D Seminar*, Budapest, Hungary, 2003.

[16] D. Gianazza, "Forecasting workload and airspace configuration with neural networks and tree search methods," *Artificial Intelligence*, vol. 174, no. 7-8, pp. 530–549, 2010.

[17] M. Xiao, J. Zhang, K. Cai, and X. Cao, "ATCEM: a synthetic model for evaluating air traffic complexity," *Journal of Advanced Transportation*, vol. 50, no. 3, pp. 315–325, 2016.

[18] D. Delahaye and S. Puechmorel, "Air traffic complexity based on dynamical systems," in *Proceedings of the 49th IEEE Conference on Decision and Control (CDC)*, pp. 2069–2074, Atlanta, GA, USA, December 2010.

[19] X. Zhu, X. Cao, and K. Cai, "Measuring air traffic complexity based on small samples," *Chinese Journal of Aeronautics*, vol. 30, no. 4, pp. 1493–1505, 2017.

[20] X. Zhu, K. Cai, and X. Cao, "A semi-supervised learning method for air traffic complexity evaluation," in *Proceedings of the 2017 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, Herndon, VA, USA, April 2017.

[21] H. Gunduz, "Deep learning-based Parkinson's disease classification using vocal feature sets," *IEEE Access*, vol. 7, pp. 115540–115551, 2019.

[22] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Mobile encrypted traffic classification using deep learning: experimental evaluation, lessons learned, and challenges," *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 445–458, 2019.

[23] H. Darvishi, D. Ciuonzo, E. R. Eide, and P. S. Rossi, "Sensor-fault detection, isolation and accommodation for digital twins via modular data-driven architecture," *IEEE Sensors Journal*, p. 1, 2020.

[24] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[26] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, USA, June 2014.

[27] Y. Endo, H. Toda, K. Nishida, and J. Ikedo, "Classifying spatial trajectories using representation learning," *International Journal of Data Science and Analytics*, vol. 2, no. 3-4, pp. 107–117, 2016.

[28] J. Donahue, L. A. Hendricks, M. Rohrbach et al., "Long-term recurrent convolutional networks for visual recognition and description," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 677–691, 2017.

[29] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt, "Sequential deep learning for human action recognition," in *Human Behavior Understanding*, pp. 29–39, Springer, Berlin, Germany, 2011.

[30] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification," in *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pp. 2267–2273, Austin, TX, USA, January 2015.

[31] H. Liu, Y. Lin, Z. Chen, D. Guo, J. Zhang, and H. Jing, "Research on the air traffic flow prediction using a deep learning approach," *IEEE Access*, vol. 7, pp. 148019–148030, 2019.

[32] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015, http://arxiv.org/abs/1409.1556.

[33] 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, 2015.

[34] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 187–197, 2015.

Hindawi
Complexity
Volume 2021, Article ID 2352185, 13 pages
https://doi.org/10.1155/2021/2352185

Research Article

DefogNet: A Single-Image Dehazing Algorithm with Cyclic Structure and Cross-Layer Connections

Suting Chen,[1] **Wenhao Fan**,[1] **Shaw Peter**,[1] **Chuang Zhang**,[1] **Kui Chen**,[1] **and Yong Huang**[2]

[1]*Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAEET), Nanjing University of Information Science & Technology, Nanjing 210044, China*
[2]*The Anhui Province Meteorological Science Research Institute, Hefei 230061, China*

Correspondence should be addressed to Suting Chen; sutingchen@nuist.edu.cn

Received 31 July 2020; Revised 22 October 2020; Accepted 27 November 2020; Published 12 January 2021

Academic Editor: Zhijie Wang

Inspired by the application of CycleGAN networks to the image style conversion problem Zhu et al. (2017), this paper proposes an end-to-end network, DefogNet, for solving the single-image dehazing problem, treating the image dehazing problem as a style conversion problem from a fogged image to a nonfogged image, without the need to estimate a priori information from an atmospheric scattering model. DefogNet improves on CycleGAN by adding a cross-layer connection structure in the generator to enhance the network's multiscale feature extraction capability. The loss function was redesigned to add detail perception loss and color perception loss to improve the quality of texture information recovery and produce better fog-free images. In this paper, the novel Defog-SN algorithm is presented. This algorithm adds a spectral normalization layer to the discriminator's convolution layer to make the discriminant network conform to a 1-Lipschitz continuum and further improve the model's stability. In this study, the experimental process is completed based on the O-HAZE, I-HAZE, and RESIDE datasets. The dehazing results show that the method outperforms traditional methods in terms of PSNR and SSIM on synthetic datasets and Avegrad and Entropy on naturalistic images.

1. Introduction

Images collected under fog, haze, and other weather conditions often suffer from low contrast, unclear scenes, and large color errors, which are prone to adversely affect the application of computer vision algorithms such as target detection and semantic segmentation. Therefore, the method of dehazing a single-image directly without using any a priori information is of great significance to the field of computer vision. At present, the standard dehazing methods can be divided into three according to different principles: firstly, image enhancement techniques. These methods focus mainly on the contrast of the image itself and other information. Secondly, image restoration methods are based on the typical physics models; these methods are primarily through a priori knowledge and physics models to complete the dehazing operation. Thirdly, neural network-based dehazing methods mainly use neural networks to complete the haze feature extraction, thereby completing the dehazing process.

Image enhancement-based dehazing methods focus mainly on the haze image contrast, edge gradient, and other information. The most common dehazing methods include wavelet transform [1], Retinex method [2], and histogram equalization [3]. Jobson et al. [4] proposed a defogging method for the Retinex image enhancement algorithm. Tan [5] proposed a dehazing method that maximizes the local contrast of the image. Moreover, the result obtained using dehazing has excessive saturation and loses much detailed information. Kim [6] proposed a subblock partial overlap method based on the idea of local equilibrium; nevertheless, there is a blocking artifact in this method's results. Zuiderveld [7] proposed a contrast limited adaptive histogram equalization (CLAHE) method to address this problem by adaptively limiting the images' contrast. The information, such as the contrast of the haze image, reflects the severity of

the haze to a certain extent. Still, the method for this kind of intuitive information cannot investigate the haze image formation mechanism. Thus, it often loses detailed information in the process of dehazing. This limitation makes it difficult for this technique to achieve an acceptable defogging effect.

The physical model-based image recovery methods study the foggy images from the imaging mechanism. The dehazing operation is accomplished through a combination of a priori knowledge and the proposed physical model, albeit the priori information must be estimated. The most common methods include dehazing algorithms based on image depth information [8], dehazing algorithms based on atmospheric light polarization [9], and dehazing methods based on various types of a priori information [10–13]. Mccartney first proposed the atmospheric physics model in 1975 [14]. Oakley and Satherley [15] proposed a transmission valuation method for image pixels around this theoretical model; however, it requires information such as the depth of field of the image, which is not very practical. According to the polarization properties of atmospheric light, Schechner et al. [16] estimated the depth of field information for dehazing by setting the polaroid glass. Narasimhan and Nayar [17] proposed a multiscale dehazing algorithm and collected images of the same location at different times for comparison and dehazing. Tarel and Hautiere [18] estimated the atmospheric scattering function by using median filtering, but this method produces a Halo effect when the scene is transformed. Fattal [19] estimated the transmittance by using independent component analysis, but again, this method is not suitable for dense foggy weather. He et al. [10] introduced the dark-channel prior method, which is ideal for defogging and easy to implement; however, some areas with intense light, such as the sky, may cause significant interference to this method. The emergence of dark channels provides new ideas for image defogging. To solve the problems encounted in the dark-channel a priori method, many image defogging methods based on the dark-channel a priori theory have subsequently been produced [20–22].

In recent years, CNN-based image defogging methods have become the focus of research. These methods mainly use neural networks to learn haze image features. Cai et al. [23] proposed the end-to-end DehazeNet system, using a convolutional neural network to extract the haze features to optimize transmittance estimation. A multiscale depth haze-removal network (MSCNN) was proposed by Ren et al. [24] to directly study and estimate the relationship between transmittance and foggy image, solving the shortcomings of artificial features. Zhang et al. [25] improved the edge and detailed information of the image by constructing a pyramid densely connected dehazing network that jointly optimizes the transmittance and atmospheric light values. Goodfellow et al. [26] proposed a new-framework generative adversarial network (GAN), for estimating generative models according to the fixed-point theorem through the adversarial process. GANs [26] can make the distribution produced by the generator as close as possible to the distribution of real data. GANs [26] provided a theoretical basis for new methods of neural network dehazing. Many approaches to optimize the GANs algorithm have been proposed. The DCGAN model of Radford et al. [27] introduced CNNs into the structure of GANs [26]. This model completes the feature extraction operation through CNNs and obtains a higher stability when generating high-quality samples. Conditional GAN (cGAN) proposed by Mirza and Osindero [28] adds conditional information y to the GANs [26]; this improves the stability of the model and enhances the generator's expression capabilities. Isola et al. [29] proposed the pix2pix algorithm to use cGAN [28] to learn the mapping from input to output to complete various image conversion tasks. These methods have authenticity requirements for the foggy image and its corresponding nonfog image, the data requirements are high, and the acquisition is difficult.

Based on GAN [26], Zhu et al. [30] proposed CycleGAN consisting of two unidirectional GANs. CycleGAN [30] suggests using cycle-consistent loss, which is mainly used to transform image styles and effectively overcomes the lack of constraint in the generated images. Albeit the CycleGAN [30] model suffers from noise and loss of texture details when completing the generation task for images with different texture complexity. The DefogNet network proposed in this paper mainly adds detail perception loss and color perception loss to CycleGAN [30]. It fully optimizes the CycleGAN [30] structure using cross-layer connections and the Defog-SN algorithm. Finally, it builds an end-to-end network without considering a priori information and without pairwise datasets.

The main contributions of this paper include the following:

Incomplete feature extraction, inefficient demist processing, and the need for a complex physical model are common issues of using an a priori physical model. To address this, we add the design of cross-layer connections in the generator, enhance the multiscale feature extraction capability of the model with feature pyramids, and obtain higher quality texture details of the generated images. By doing these modifications, we omit the manual design of the a priori model. Consequently, our approach requires neither fuzzy and real image samples nor any atmospheric scattering model parameters in the training and testing phases.

This study designed unique loss functions: detail perception loss and color perception loss to optimize DefogNet for single-image dehazing and to address the color shifts and high contrast resulting from the defogging operation.

We introduce spectral normalization in the discriminative network and propose the Defog-SN algorithm, which has strong generalization ability and effectively solves the problem of insufficient diversity of generated samples and improves the quality of defogged images enhancing the overall stability of the network as well as the convergence speed.

The rest of this paper is organized as follows: Section 2 mainly describes the methods proposed in this paper.

Section 3 draws relevant experimental conclusions and analyzes them in depth. Section 4 concludes the article.

2. Proposed Method

2.1. Structure of DefogNet. This paper presents a DefogNet-based single-image dehazing method that improves on the CycleGAN [30] network architecture. CycleGAN [30] adds cycle-consistent loss; the central role of cycle consistency loss is to extract the combination of high-level and low-level features in the VGG16 architecture [31], which creates constraints on the generator and preserves the original image structure. CycleGAN [30] can accurately calculate the $L1$ paradigm of the original image and the loop-through images to perform the task of unpaired image-to-image conversion. Still, the loss between the original image and the loop-through image does not allow the complete texture information to be recovered.

DefogNet is an improved version of CycleGAN [30], the structure of which is shown in Figure 1. In this paper, we propose the Defog-SN algorithm and add cross-layer connections to the generator, which optimizes the overall performance of the network and improves the quality of dehazed images. DefogNet redesigns the loss function and introduces detail perception loss and color perception loss biased on cycle-consistent loss. It then uses this function to calculate the total loss of the model. In this way, it preserves the initial information more completely in image reconstruction, effectively optimizing the dehazing function.

DefogNet consists of two generators, G and F, and two discriminators, D_X and D_Y. The two generative adversarial networks compete with each other as well as constraints. By training simultaneously, they learn the foggy features while keeping the image background structure almost unchanged and use unpaired datasets without manual labeling. This unsupervised approach dramatically simplifies the work in the data preparation stage.

In this paper, the foggy image dataset and the clear image dataset are used to train the model. The foggy image data and clear image data are defined as domain X and domain Y. There is no correspondence between the images in domain X and domain Y. After inputting the image in X into the generating model G, G will generate a new image Y_G. Hence, Y_G and the image in Y serve as the input to the discriminating model D_X, which determines when the input image is from the real clear data Y or the pseudoimage generated by the generator G. The result is fed back to G and used to strengthen G. Under the guidance of D_X, Y_G continuously reduces the gap between Y_G and the image in the clear data set Y.

Similarly, generator F acts to reduce Y_G and Y to foggy images and Y_G and Y are fed into the 2nd generating model F to generate a new image X_F, which is designed to make X_F as similar as possible to the image in X through a loss function to ensure that the learned mapping is meaningful. This model's cyclic structure allows the two GANs to generate increasingly realistic images, i.e., images in the fogged dataset X to complete the dehazing.

2.2. Cross-Layer Connection Structure on Generator. To properly train the sample distribution and effectively optimize the quality of the final generated image, the network structure was chosen was an encoder-transition-decoder. The role of the first part of the encoder layer is to complete the feature extraction process. The second part of the transition layer is to combine the image's different features through the residual network and reorganize the features. The third part of the decoder layer is to reconstruct the image.

When comparing the fogged image and the defogged image, it is found that the fogged image and the defogged image should have the same background, similar spatial structure, and details of the object. So, we need to preserve some of the input image structure, share some of the information between the input and output, and send this information directly to the decoding layer without going through the transformation layer. Based on this, our approach improves the generator structure and designs a codec network with a cross-layer connection structure to break the bottleneck of information loss during the codec process and discards the method of merely connecting all channels of the symmetric layer, as shown in Figure 2. The output of each convolution layer in the encoder will be directly inputted to the corresponding decoder simultaneously with the inverse convolution result of the output of the next convolution layer. This structure keeps the feature map's size consistent, effectively reducing the difference between the output and the original input; otherwise, the features of the original image will not be retained in the output, and the output will deviate from the background contour of the original image.

In this paper, the batch normalization layer is removed for each unit in the transformation layer, and the SeLU activation function [32] is used to automatically normalize the sample distribution to a mean of zero and a standard deviation of one. The relevant formula is

$$\operatorname{selu}(x) = \lambda \begin{cases} x, & x > 0, \\ \alpha e^{x} - \alpha, & x \leq 0. \end{cases} \tag{1}$$

The structure can ensure the high consistency between the defogged image and the original banded fog image except for haze and improve the multiscale fusion feature extraction capability of the network for haze.

2.3. Defog-SN Algorithm on Discriminator. CycleGAN [30] consists of two unidirectional GANs, which have the disadvantages of unstable training and collapse-prone models due to the discriminant network's poor control performance. In this paper, the Defog-SN algorithm is to solve this problem. This algorithm incorporates spectral normalization [33] into the discriminant network by inserting a spectral normalization layer into each convolutional layer to optimize the quality of dehazed images. The discriminator mainly consists of 6 layers; the first four layers are used to complete the input image's feature extraction process and then the last layer. The last fully-connected layer outputs Wasserstein distance. The discriminator structure is shown in Figure 3.

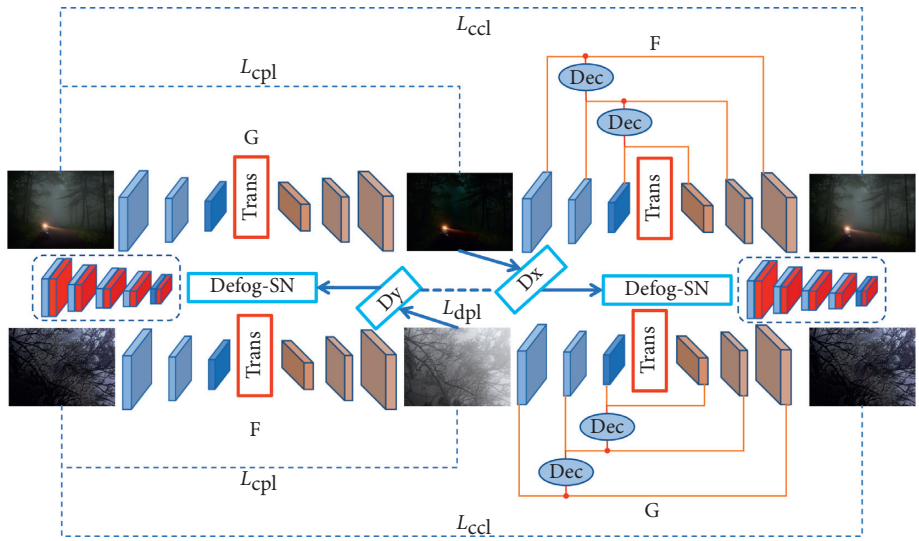


Figure 1: DefogNet network structure.

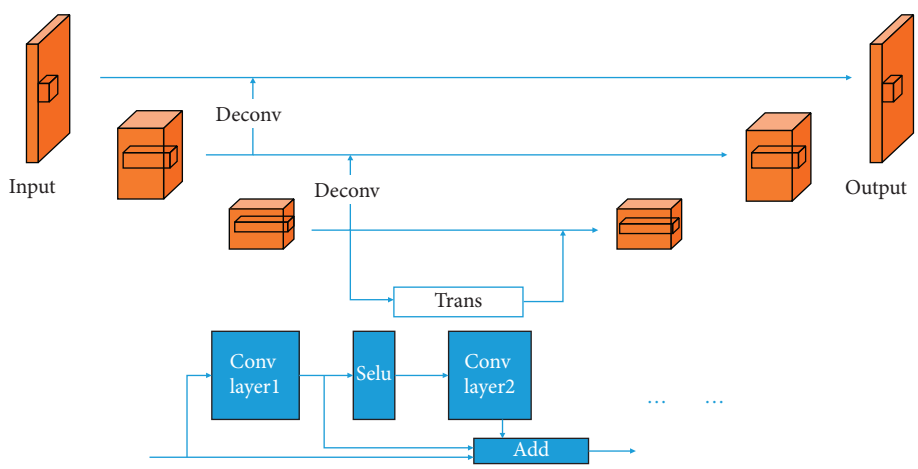


Figure 2: Structure of the generator.

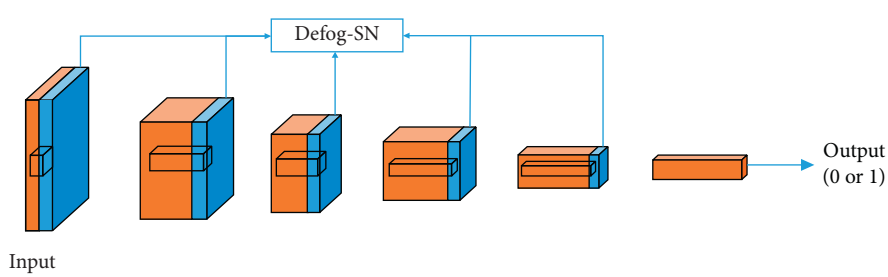


Figure 3: Discriminator structure.

The GAN stability theorem states that if the discriminant network can conform to a 1-Lipschitz continuum in terms of the input, the output, and the control of the discriminant network can be significantly optimized. The stability of the GAN training process can be further enhanced [34]. According to the Lipschitz theory of complex functions, typical neural networks consist of multilayer structures, which can be viewed as a more complicated complex function. If the individual functions can satisfy the 1-Lipschitz continuum, then the composite functions that combine these functions also satisfy the continuum. DefogNet's discriminant network's activation functions are all Leaky

ReLU [35] functions, which satisfy 1-Lipschitz continuity, so, as long as each convolutional layer of the discriminant network satisfies 1-Lipschitz continuity, the whole discriminant network satisfies 1-Lipschitz continuity. The formula for the Lipschitz constraint is

$$\frac{\left\|f(x)-f(x')\right\|_2}{\left\|x-x'\right\|_2} \leq \beta, \tag{2}$$

where β is a constant and the gradient of a function satisfying the formula is always limited to a range less than or equal to β. The Defog-SN algorithm uses spectral normalization to provide a global normal for the discriminant network. The maximum singularity value can have a decisive effect on the Lipschitz continuity constant of the linear operator, thus enabling the parameter matrix to use more features when generating images, thus effectively enhancing the diversity of samples and optimizing the quality of image defogging.

The Defog-SN algorithm adds spectral normalization after each convolutional layer to make each layer conform to the 1-Lipschitz continuity. The details of the algorithm are as follows: for the convolutional layer t: $h_{\text{in}} \longrightarrow h_{\text{out}}$, A refers to the matrix of convolutional layer parameters, and its spectral parameter $\sigma(A)$ is calculated by the following equation [36]:

$$\sigma(A) = \max_{h\neq 0} \frac{\|Ah\|_2}{\|h\|_2} = \max_{\|h\|_2\leq 1} \|Ah\|, \tag{3}$$

where $\sigma(A)$ is equal to the maximum singularity of matrix A and the Lipschitz continuity constant of the convolutional layer t is equal to the spectral paradigm of its convolutional layer parameter matrix. The spectral paradigm $\sigma(W)$ of the convolutional layer parameter matrix, W, is calculated so that the maximum singularity of the convolutional layer parameter matrix W_{SN} after completion of the normalization process is equal to 1. In this way, the input and output satisfy the 1-Lipschitz condition. The formula for W_{SN} is as follows:

$$W_{\text{SN}} = \frac{W}{\sigma(W)}. \tag{4}$$

The vector u is randomly initialized as the right singularity eigenvector of the parameter matrix W. The following formula calculates the left singularity eigenvector v:

$$\begin{aligned} v &= \frac{\left\|W^T u\right\|}{\left\|W^T u\right\|_2}, \\ u &= \frac{\|Wv\|}{\|Wv\|_2}. \end{aligned} \tag{5}$$

After iterations, it is possible to calculate the opening $\sqrt{\lambda_1}$ of the maximum eigenvalue of $W^T W$, i.e., the maximum singular value of the matrix W:

$$\sqrt{\lambda_1} = u^T W v. \tag{6}$$

The update of the convolutional layer parameter matrix W^i is accomplished as follows:

$$W^i \leftarrow W^i \leftarrow \alpha \nabla_{W^i}\left(W^i_{\text{SN}}\left(W^i\right), X, Y\right), \tag{7}$$

where α is the learning rate.

We use deconvolution to simplify and speed up the calculation of the spectral norm of convolution. Hanie et al. [37] developed a method to calculate all singular values, including the spectral norm; however, this method is only suitable for convolution filters with step size 1 and 0 padding. In the training process, the standardization factor depends on the step size and filling scheme of the control convolution operation.

This paper proposed an efficient method to calculate the maximum singular value (i.e., the spectral norm) of a 6-layer convolution layer with an arbitrary step size and filling scheme. The output feature map ψ of layer i in the neural network can be expressed as a linear operation of input X:

$$\psi_i(X) = \sum_{j=1}^{M} F_{i,j} * X_j, \tag{8}$$

where M is the feature map of the input and $F_{i,j}$ is a filter. Here, we ignore the additional bias term. We vectorize X and let $T_{i,j}$ express the overall linear operation related to $F_{i,j}$:

$$\psi_i(X) = \left[T_{1,1} \ \cdots \ T_{1,M}\right]X. \tag{9}$$

Then, the convolution operation can be expressed as

$$\psi(X) = \begin{bmatrix} T_{1,1} & \cdots & T_{1,M} \\ \vdots & \ddots & \vdots \\ T_{N,1} & \cdots & T_{N,M} \end{bmatrix} X = WX. \tag{10}$$

By convolution transposition, we can obtain the spectral norm associated with W. By using W^T effectively, we can implement this matrix multiplication more efficiently, without explicitly constructing W. The correlation spectrum norm $\sigma(W)$ can be obtained by the power iteration method, and appropriate step size and filling parameters are added in convolution and convolution transposition operation. We use the same value u repeatedly, and only update W once per step. We use a wider range to restrict $(W) \leq \beta$:

$$W_{SN} = \frac{W}{\max(1, (\sigma(W)/\beta))}, \tag{11}$$

which results in a faster training speed.

We now use Wasserstein distance as a criterion to measure the generated distribution p_g and the real distribution p_{data}. Due to the introduction of 1-Lipschitz continuity, we need to restrict the variation range of network parameters within a certain range; the change range of parameters should not exceed a certain constant in each update. Therefore, the Wasserstein distance between the real data distribution p_{data} and the generated data distribution p_g can be expressed as follows:

$$D_w = E_{x\sim p_{\text{data}}}\left[f_w(x)\right] - E_{x\sim p_g}\left[f_w(x)\right]. \tag{12}$$

The smaller the D_w, the more likely the generated distribution p_g is to be close to the true distribution p_{data}. Due

to the introduction of spectral normalization, the function is differentiable in all cases, allowing us to solve the problem of gradient vanishing in the GAN model's training process [38]. Therefore, the objective function of the DefogNet discriminator is as follows:

$$\mathrm{obj}^{D} = \min\left(E_{x\sim p_g}\left[f_w(x)\right] - E_{x\sim p_{\mathrm{data}}}\left[f_w(x)\right]\right). \tag{13}$$

Next, we enhance the Lipschitz constraint by gradient penalty. Firstly, we use the random sampling method to obtain the True sample X_{data}, False sample X_g, and a random number ϑ in the range of [0,1]. Then, we randomly interpolate the sample between X_{data} and X_g:

$$\widehat{X} = \vartheta X_{\mathrm{data}} + (1-\vartheta)X_g. \tag{14}$$

The distribution satisfied by $\widehat{X}$ is denoted as $p_{\widehat{X}}$, and the improved objective function of the DefogNet is

$$\mathrm{obj}^{(G,D)} = \min\left(\begin{array}{c} E_{x\sim p_g}\left[D_w(x)\right] - E_{x\sim p_{\mathrm{data}}}\left[D_w(x)\right] + \\ E_{\widehat{X}\sim p_{\widehat{X}}}\left[\left\|\nabla_{\widehat{X}} D_w(\widehat{X})\right\|_2 - \left(D_w(x) - D_w(\widehat{X})\right)^2\right]^2 \end{array}\right). \tag{15}$$

2.4. Loss Function of DefogNet. CycleGAN's [30] loss function consists of the generator and discriminator's adversarial loss function and a cycle-consistent loss function. The generated adversarial loss function consists of the discriminator's probability estimate of the true sample and the discriminator's probability estimate of the generated sample:

$$\begin{aligned}\min_G \max_D V(D,G) = {} & E_{x\sim P_{\mathrm{data}(x)}}\left[\log D(x)\right] \\ & + E_{z\sim P_{z(x)}}\left[\log\left(1 - D(G(z))\right)\right].\end{aligned} \tag{16}$$

The first step, for this stage, is to use the generator to transform the real image in domain X into a false image in domain Y. Then, use the generator to complete the reconstruction process to obtain the reconstructed image Y_G, preserving all the image's original information. Then, this and the real image Y are passed to the discriminator D_X to determine the authenticity, thereby obtaining a complete one-way GAN. The correlation loss function is

$$\begin{aligned} L_{\mathrm{GAN}}\left(G, D_Y, X, Y\right) &= E_{y\sim P_{\mathrm{data}(y)}}\left[\log D_w(y)\right] \\ &\quad + E_{x\sim P_{\mathrm{data}(x)}}\left[\log\left(1 - D_w\left(f_w(x)\right)\right)\right], \\ L_{\mathrm{GAN}}\left(F, D_X, X, Y\right) &= E_{x\sim P_{\mathrm{data}(x)}}\left[\log D_w(x)\right] \\ &\quad + E_{y\sim P_{\mathrm{data}(y)}}\left[\log\left(1 - D_w\left(f_w(y)\right)\right)\right]. \end{aligned} \tag{17}$$

The cycle-consistent loss function is introduced into the network to learn the mapping of $G_{X\longrightarrow Y}$ and $F_{Y\longrightarrow X}$ and can convert X to Y and back again successfully, thus avoiding that all images are mapped to the same image in Y:

$$L_{\mathrm{CCL}} = \left\|\phi(x) - \phi(F(G(x)))\right\|_2^2 + \left\|\phi(y) - \phi(G(F(y)))\right\|_2^2. \tag{18}$$

The CycleGAN [30] loss function is

$$L_{\mathrm{CYC}} = L_{\mathrm{GAN}}\left(G, D_Y, X, Y\right) + L_{\mathrm{GAN}}\left(F, D_X, X, Y\right) + \gamma L_{\mathrm{CCL}}, \tag{19}$$

where X and Y refer to the two data domains, and x and y are the above data domains within the sample data, G refers to the X to Y mapping function, F refers to the Y to X mapping function, D_X and D_Y refer to the discriminator, and γ is the weight of the cycle-consistent loss.

The least-squares loss method used by CycleGAN [30] penalizes the outlier samples too much, which reduces the diversity of the generated samples. A single loss function does not guarantee that the model will successfully map a single input X_i to the expected output Y_i. This study adds color perception loss function and detail perception loss function based on the original loss function to train the network more optimally on unpaired images. The loss is mainly used to estimate the image's differences after dehazing and minimize the change to the original image. Discriminator G:

$$L_{\mathrm{dpl}}\left(G_{X\longrightarrow Y}\right) = E_{y\sim P_{\mathrm{data}(y)}}\left[D_X(F(y)) - 1\right]^2 + \frac{1}{2}E_{y\sim P_{\mathrm{data}(y)}}\sqrt{\left[F(y) - 1\right]^2 - y^2}. \tag{20}$$

Discriminator F:

$$L_{\mathrm{dpl}}\left(F_{Y\longrightarrow X}\right) = E_{x\sim P_{\mathrm{data}(x)}}\left[D_Y(G(x)) - 1\right]^2 + \frac{1}{2}E_{x\sim P_{\mathrm{data}(x)}}\sqrt{\left[G(x) - 1\right]^2 - x^2}. \tag{21}$$

We combine the equations to form the detail perception loss.

$$L_{\text{dpl}} = L_{\text{dpl}}\left(F_{X\longrightarrow Y}\right) + L_{\text{dpl}}\left(G_{Y\longrightarrow X}\right). \tag{22}$$

Because the dehazing process must be completed for r, g, b three types of channels to complete the operation, but need to maintain the image after the completion of defogging does not produce large color differences. Therefore, it is necessary to add the color perception L_{cpl} when generating the image:

$$L_{\text{cpl}}(I) = \sum_{w=1}^{W}\sum_{h=1}^{H}\begin{cases} \cos^{-1}\left\{\dfrac{0.5[(r-g)+(r-b)]}{\left[(r-g)^2+(r-b)(g-b)\right]^{1/2}}\right\}, & b \le g, \\ 2\pi - \cos^{-1}\left\{\dfrac{0.5[(r-g)+(r-b)]}{\left[(r-g)^2+(r-b)(g-b)\right]^{1/2}}\right\}, & b > g. \end{cases} \tag{23}$$

W and H are the width and height of the image. The final loss function is

$$L_{\text{defog}} = L_{\text{dpl}} + L_{\text{cpl}}(I) + L_{\text{CYC}}. \tag{24}$$

3. Experiences and Results

In the experimental section, the method will be compared with the results of several advanced methods for hazing, including CycleGAN [30], providing qualitative and quantitative analysis of the experimental results on both synthetic and naturalistic datasets. This experiment completes all training as well as testing procedures on TensorFlow [39]. The NVIDIA Tesla V100 GPU is used for model training, optimized using the Adadelta algorithm [40] with a good adaptive learning rate, a batch size of 1, and a learning rate of $2 * 10^{-4}$.

3.1. Datasets. Experiments were conducted on the I-HAZE [41], O-HAZE [42], and RESIDE [43] datasets. The outdoor dataset of RESIDE [43] contains 8970 clear images and 31,950 foggy images synthesized from clear images. O-HAZE [42] is an outdoor scene database containing 45 pairs of realistic foggy images of outdoor scenes and the corresponding nonfoggy images. I-HAZE [41] contains 35 pairs of real indoor scenes with fog and the corresponding nonfog images. I-HAZE [41] and O-HAZE [42] are common experimental data sets of dehazing scenes taken from controllable scenes made by professional fogging machines, which have similar lighting conditions. This experiment randomly selects 4900 foggy and nonfog images from I-HAZE [41], O-HAZE [42], and RESIDE [43] for training and validation. To facilitate the comparison with existing advanced methods, PSNR and SSIM are used as evaluation metrics in this paper to compare the synthetic target test set containing 400 indoor images and 400 outdoor images. AveGrad and Entropy are selected as evaluation metrics for evaluating the method's performance on natural and realistic images, and experiments are conducted on the RTTS dataset in RESIDE [43]. The photos needed to be resized to 256×256 in size when imported to the network.

3.2. Validation of DefogNet Method. To verify the performance of the generator's improved cross-layer connection structure, we compare the DefogNet model using different generator structures under the same conditions and compare the network using the DefogNet without a cross-layer connection structure. The experiment is set to $300^{*}10^{3}$ iterations. The network model is saved and output after every $5^{*}10^{3}$ training iterations, and the PSNR index evaluation of the model in the RESIDE data set is performed. The results show that the model using the cross-layer connection structure has the highest PSNR evaluation index in the experimental iterative-training process compared with the original model. The comparison results are shown in Figures 4 and 5.

For the neural network based on Gan, the network's convergence speed and stability are important indexes to evaluate its performance. To verify the performance of DefogNet, we train the model processed by Defog-SN to compare with those without this processing and compare the convergence speed of DefogNet's discriminator and CycleGAN's discriminator. In the process of $300 * 10^{3}$ iterations, our method's convergence speed is higher than that of other methods, which also indicates the effectiveness of Defog-SN in improving the network performance. The comparison results are shown in Figure 6.

Several models with different loss functions are trained under the same conditions to verify the multi-loss function's performance. One-hundred images were randomly selected from RESIDE [43] for the experiment. Defognet (NET-4) is compared with different combination loss models in Table 1. The design of the performance verification model is shown in Table 1. The results show that the model using multiple loss fusion has the highest PSNR evaluation index in the $300 * 10^{3}$ iterations training process compared with the model using other loss functions. The comparison results are shown in Figures 7 and 8.

3.3. Results on Synthetic Datasets. For RESIDE [43], I-HAZE [41], and O-HAZE [42] datasets, we choose PSNR and SSIM as the quantitative evaluation indexes for the experimental results. The posttraining evaluation of this paper's algorithm

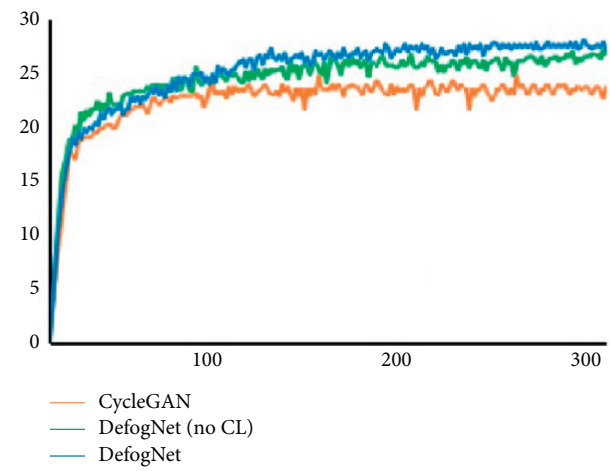


Figure 4: The convergence of PSNR values.

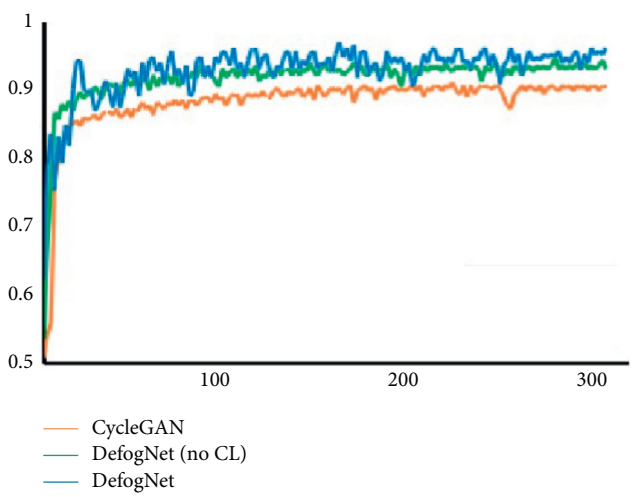


Figure 5: The convergence of SSIM values.

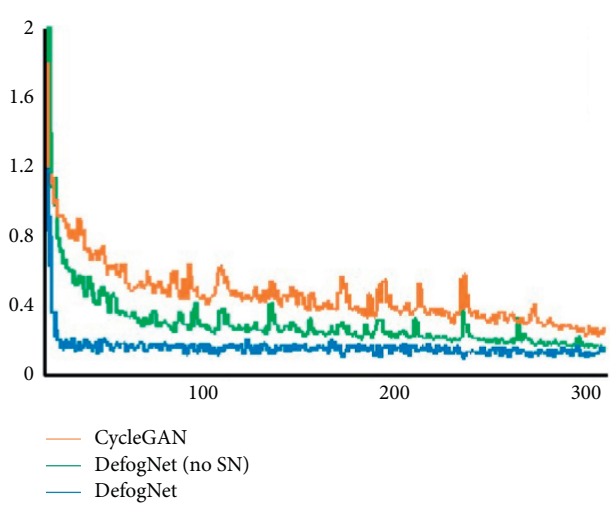


Figure 6: The gradient convergence of discriminator.

Table 1: Neural networks with different loss functions.

Models	Loss function
Net-1	$L_{\mathrm{CYC}} + L_{\mathrm{cpl}}(I)$
Net-2	$L_{\mathrm{dpl}} + L_{\mathrm{CYC}}$
Net-3	$L_{\mathrm{dpl}} + L_{\mathrm{cpl}}(I)$
Net-4	$L_{\mathrm{dpl}} + L_{\mathrm{CYC}} + L_{\mathrm{cpl}}(I)$

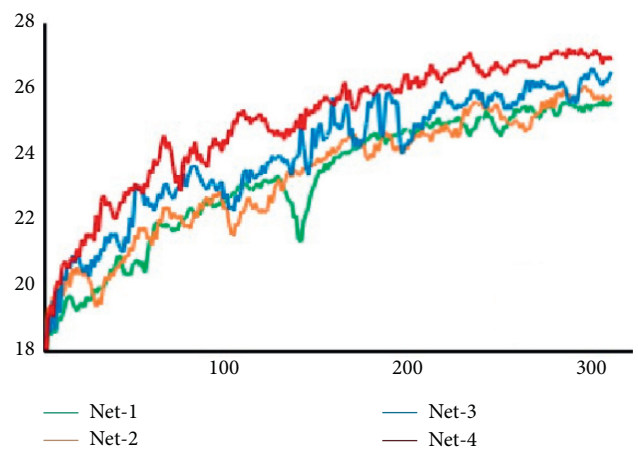


Figure 7: Comparison of PSNR values using different loss function models.

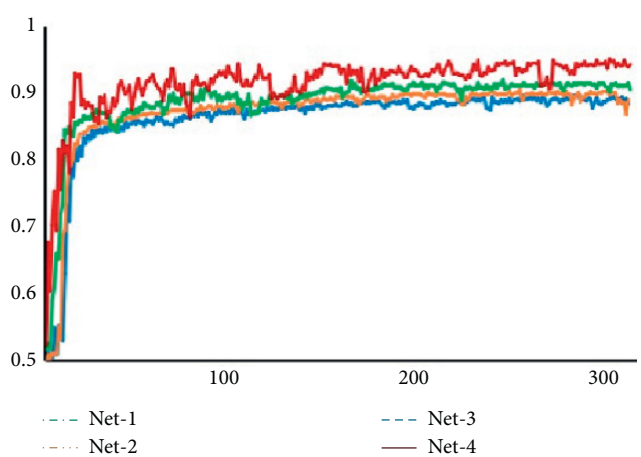


Figure 8: Comparison of SSIM values using different loss function models.

is compared with several advanced single-image dehazing methods, as well as the original CycleGAN effect under the same parameter setting environment, and the results are shown in Table 2 and Figure 9. It can be seen that the dehazing results produced by this algorithm have higher PSNR and SSIM values (Figures 9).

According to the method proposed in this paper, the PSNR and SSIM values of the results are compared with those of CycleGAN and other methods. It can be found that DefogNet's structural optimization method is useful in

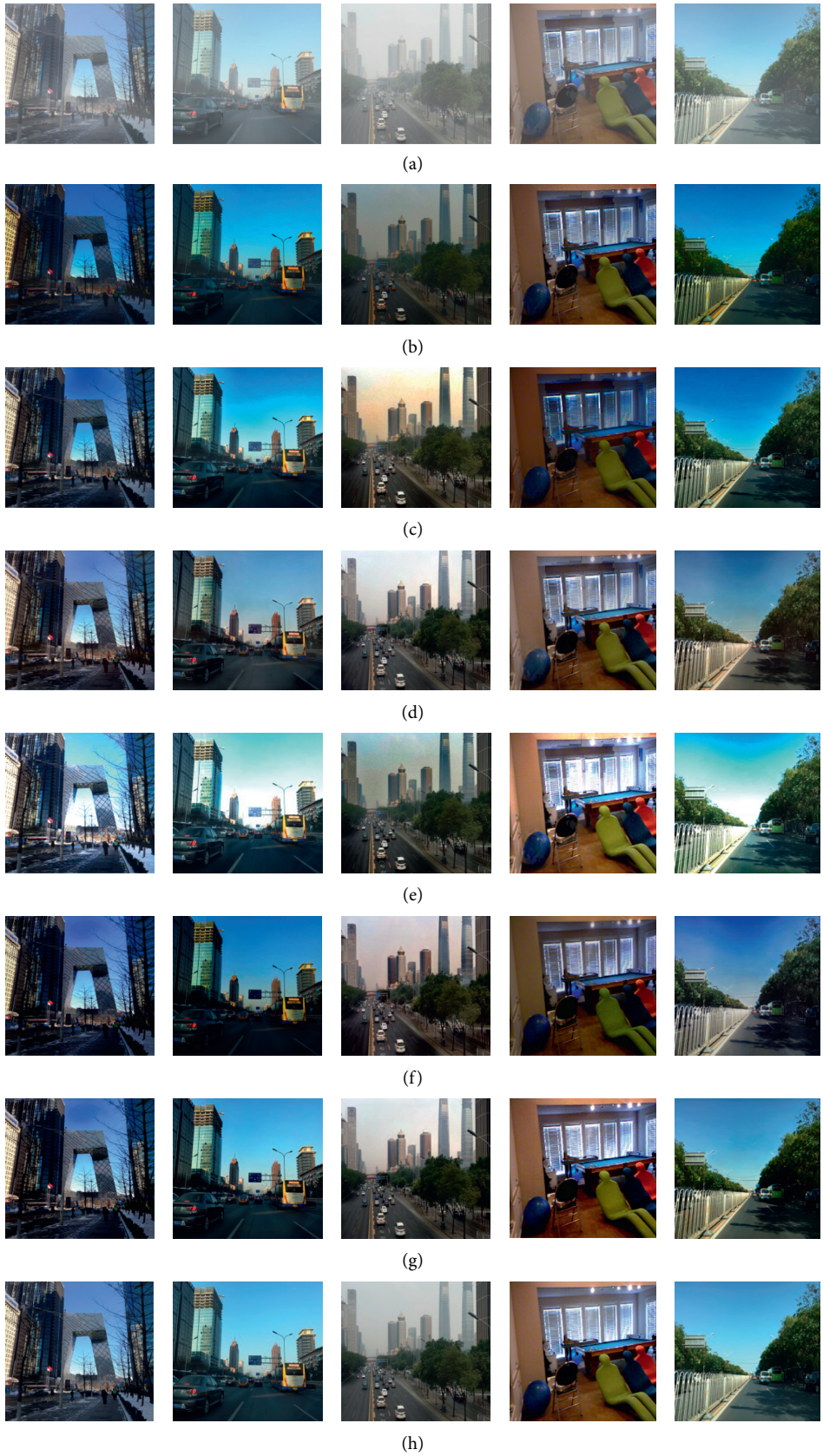


Figure 9: Comparison of results on synthetic datasets: (a) Haze, (b) Cai [23], (c) He [10], (d) Zhang [25], (e) Ren [24], (f) Zhu [30], (g) this study, and (h) GT.

(a)

(b)

(c)

(d)

(e)

(f)

Figure 10: Continued.

(g)

Figure 10: Comparison of results on natural, realistic images. (a) Haze, (b) Cai [23], (c) He [10], (d) Zhang [25], (e) Ren [24], (f) Zhu [30], and (g) Our study.

Table 2: Quantitative comparisons of image dehazing on the I-HAZE, O-HAZE, and RESIDE data sets.

		Cai [23]	He [10]	Zhang [25]	Ren [24]	Zhu [30]	Ours
First image	PSNR	19.13	15.66	24.57	18.22	23.12	29.45
	SSIM	0.79	0.84	0.88	0.85	0.89	0.95
Second image	PSNR	20.86	19.45	25.10	20.43	25.72	27.95
	SSIM	0.81	0.84	0.89	0.82	0.88	0.93
Third image	PSNR	21.96	18.66	23.18	19.77	22.28	28.31
	SSIM	0.85	0.79	0.82	0.86	0.85	0.96
Fourth image	PSNR	23.50	20.07	26.13	22.79	24.33	29.37
	SSIM	0.79	0.80	0.87	0.83	0.86	0.91
Fifth image	PSNR	22.48	21.09	23.64	22.17	23.94	26.95
	SSIM	0.83	0.83	0.86	0.85	0.85	0.94

substantially improving the performance of various aspects of the original CycleGAN. Although He's method [10] can remove some haze, it produces artifacts and color distortion, especially in the sky and light-colored areas. Cai's method [23] and Ren's method [24] need to estimate the transmittance, and due to inaccurate transmittance estimation, the defogging results still contain more artifacts and haze residue. Zhang's method can generate clear images; however, compared with our algorithm's dehazing images, there are more unclear object outlines. The overall image color is darker, and the sky's color in the images is slightly distorted. Zhu's method [30] effectively avoids artifacts but suffers from color shifts and distortions. These methods have more or less color distortion. Our method has achieved high PSNR and SSIM values and sound visual effects. It also indicates that this article's loss function has an improved impact on color constraints. This algorithm avoids estimating the transmittance and atmospheric light values and produces sharper dehazed images than other algorithms. Not only does the method obtain higher quality evaluation scores, but the defogging results also yield sharper image edge details and less noise while avoiding color distortion problems (Table 2).

3.4. Results on Natural Realistic Images. At present, the neural network used for the image dehazing problem is challenging to achieve good results on natural real image data sets. Because CNN tends to have overfitting problems, it is more likely to solve the task for a specific scene or a particular dataset. This paper chose a cross-data set method for experimental analysis in the training and testing stage and selected an everyday, natural, image data set to compare with other methods' dehazing results. Since there is no original image for comparison, the information entropy (Entropy) and average gradient (AveGrad), which reflect the image clarity, are used as the evaluation criteria for the defogging results in this paper. The dehazing results of this method and other methods, such as CycleGAN [30], are presented for comparison. Figure 10 shows the foggy images of three real scenes and the corresponding dehazing results generated by several algorithms.

From the data in Table 3, it follows that the gradient value of the original foggy images is low, most of the edge details of buildings or plants in the images were obscured by the haze, and the images are not clear (i.e., the image information entropy value is low). In comparison, the gradient and entropy values of He's algorithm [10] and Zhang's algorithm [25] are relatively large. Still, the color of the sky part of the image (i.e., the upper part of the image) is more distorted in He's algorithm. Due to the shortcomings of the dark-channel prior algorithm, there is still haze residue and dark color in the dehazed image. Although Cai's algorithm [23] and Ren's algorithm [24] have relatively large Entropy values, there is still haze residue in the image after dehazing, which affects the detailed display of objects in the image. Zhu's algorithm [30] achieves a high-quality score, but it is challenging to ensure precise object edges and details, and the overall color of the image is not natural enough. Compared with other algorithms, the gradient value and information entropy of the algorithm in this paper were larger than other algorithms. The details in the image are

Table 3: Quantitative comparisons of image dehazing on the RTTS dataset from RESIDE.

		Cai [23]	He [10]	Zhang [25]	Ren [24]	Zhu [30]	Ours
First image	AveGrad	5.8524	6.0103	6.5384	5.4105	6.3822	6.6396
	Entropy	7.2023	6.925	7.4511	7.1409	7.2946	7.3975
Second image	AveGrad	6.2126	6.4438	6.7374	5.8781	6.7822	6.9413
	Entropy	7.2384	7.7843	7.8394	7.4974	7.7908	7.8233

retained, and the clarity is higher. The overall color of the dehazing result is more natural, which indicates that this method can effectively solve the overfitting problem for similar data.

4. Conclusion

This paper proposes an effective new defogging method DefogNet. This method is optimized based on the CycleGAN method, completely omits artificially extracting features. and does not require scene prior information. It is a method with a wide range of adaptations. The optimized network architecture and loss function in this study solve the problem of difficult training sample collection encountered when using deep learning methods for dehazing research, greatly reducing the difficulty of obtaining training datasets, making this method more practical and accurate with other deep learning-based dehazing method. This paper adds cross-layer connections in the generator, which optimizes the high and low-level fusion feature extraction capability of the model, effectively avoiding overfitting and improving the generated images' quality. A unique loss function is designed to add detail perception loss and color perception loss to avoid the color differences and reconstruction loss in the image caused by the dehazing operation, effectively improving the restoration of the image's details after dehazing. The Defog-SN algorithm is proposed to improve the structure of the discriminator so that the entire discriminant network satisfies the 1-Lipschitz continuum, thus enhancing the stability of the model and avoiding the problem that the GANs model is prone to collapse. Furthermore, experimental results produced using natural image datasets demonstrate the generality of the present defogging method for images of different scenes.

Data Availability

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

Conflicts of Interest

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

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No.61906097) and a project funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

References

[1] Y. Chen, D. Li, and J. Q. Zhang, "Complementary color wavelet: a novel tool for the color image/video analysis and processing," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 1, pp. 12–27, 2017.

[2] M. Yamakawa and Y. Sugita, "Image enhancement using Retinex and image fusion techniques," *Electronics and Communications in Japan*, vol. 101, no. 8, pp. 52–63, 2018.

[3] A. Faed, E. Chang, M. Saberi, O. K. Hussain, and A. Azadeh, "Intelligent customer complaint handling utilising principal component and data envelopment analysis (PDA)," *Applied Soft Computing*, vol. 47, pp. 614–630, 2016.

[4] D. J. Jobson, Z. Rahman, and G. A. Woodell, "A multiscale retinex for bridging the gap between color images and the human observation of scenes," *IEEE Transactions on Image Processing*, vol. 6, no. 7, pp. 965–976, 1997.

[5] R. T. Tan, "Visibility in bad weather from a single image," in *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, IEEE, Anchorage, AK, USA, June 2008.

[6] Y. T. Kim, "Contrast enhancement using brightness preserving bi-histogram equalization," *IEEE Transactions on Consumer Electronics*, vol. 43, no. 1, pp. 1–8, 1997.

[7] K. Zuiderveld, "Contrast limited adaptive histogram equalization," *Graphics Gems*, pp. 474–485, 1994.

[8] J. Kopf, B. Neubert, B. Chen et al., "Deep photo," *ACM Transactions on Graphics*, vol. 27, no. 5, pp. 1–10, 2008.

[9] Y. Y. Schechner, S. G. Narasimhan, and S. K. Nayar, "Instant dehazing of images using polarization," in *Proceedings of the 2001 IEEE computer Society Conference on computer vision and Pattern recognition, CVPR 2001*, IEEE, Kauai, HI, USA, December 2001.

[10] K. He, J. Sun, and X. Tang, "Single image haze removal using dark channel prior," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 12, pp. 2341–2353, 2010.

[11] Q. Zhu, J. Mai, and L. Shao, "A fast single image haze removal algorithm using color attenuation prior," *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 3522–3533, 2015.

[12] C. O. Ancuti, C. Ancuti, and C. Hermans, "A fast semi-inverse approach to detect and remove the haze from a single image," in *Proceedings of the Asian Conference on Computer Vision*, pp. 501–514, Springer, Berlin, Heidelberg, December 2010.

[13] D. Berman and S. Avidan, "Non-local Image dehazing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1674–1682, IEEE, Las Vegas, NV, USA, June 2016.

[14] E. J. McCartney, "Optics of the atmosphere: scattering by molecules and particles," *IEEE Journal of Quantum Electronics*, vol. 14, no. 9, pp. 698-699, 1976.

[15] J. P. Oakley and B. L. Satherley, "Improving image quality in poor visibility conditions using a physical model for contrast degradation," *IEEE Transactions on Image Processing*, vol. 7, no. 2, pp. 167–179, 1998.

[16] Y. Y. Schechner, S. G. Narasimhan, and S. K. Nayar, "Polarization-based vision through haze," *Applied Optics*, vol. 42, no. 3, pp. 511–525, 2003.

[17] S. G. Narasimhan and S. K. Nayar, "Interactive (de) weathering of an image using physical models," in *Proceedings of the Methods in Computer IEEE Workshop on Color and Photometric Vision*, IEEE, Nice, France, October 2003.

[18] J. P. Tarel and N. Hautiere, "Fast visibility restoration from a single color or gray level image," in *Proceedings of the 2009 IEEE 12th International Conference on Computer Vision*, pp. 2201–2208, IEEE, Kyoto, Japan, September 2009.

[19] R. Fattal, "Single image dehazing," *ACM Transactions on Graphics*, vol. 27, no. 3, pp. 1–9, 2008.

[20] S. C. Pei and T. Y. Lee, "Nighttime haze removal using color transfer pre-processing and dark channel prior," in *Proceedings of the 2012 19th IEEE International Conference on Image Processing*, pp. 957–960, IEEE, Orlando, FL, USA, September 2012.

[21] Y. Shuai, R. Liu, and W. He, "Image haze removal of wiener filtering based on dark channel prior," in *Proceedings of the 2012 Eighth International Conference on Computational Intelligence and Security*, pp. 318–322, IEEE, Guangzhou, China, November 2012.

[22] D. Fan, X. Guo, and X. Lu, "Image defogging algorithm based on sparse representation," *Complexity*, vol. 2020, Article ID 6835367, , 2020.

[23] B. Cai, X. Xu, K. Jia, C. Qing, and D. Tao, "DehazeNet: an end-to-end system for single image haze removal," *IEEE Transactions on Image Processing*, vol. 25, no. 11, pp. 5187–5198, 2016.

[24] W. Ren, S. Liu, and H. Zhang, "Single image dehazing via multiscale convolutional neural networks," in *Proceedings of the European Conference on Computer Vision*, pp. 154–169, Springer, Amsterdam, The Netherlands, October 2016.

[25] X. Zhang, H. Dong, and Z. Hu, "Gated fusion network for joint image deblurring and super-resolution," 2018, https://arxiv.org/abs/1807.10806.

[26] I. Goodfellow, J. Pouget-Abadie, and M. Mirza, "Generative adversarial nets," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2672–2680, Montreal, Canada, December 2014.

[27] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, https://arxiv.org/abs/1511.06434.

[28] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014, https://arxiv.org/abs/1411.1784.

[29] P. Isola, J. Y. Zhu, and T. Zhou, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1125–1134, IEEE, Honolulu, HI, USA, July 2017.

[30] J. Y. Zhu, T. Park, and P. Isola, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2223–2232, IEEE, Venice, Italy, October 2017.

[31] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, https://arxiv.org/abs/1409.1556.

[32] G. Klambauer, T. Unterthiner, and A. Mayr, "Self-normalizing neural networks," in *Proceedings of the Advances in neural Information Processing Systems*, pp. 971–980, IEEE, Long Beach, CA, USA, December 2017.

[33] T. Miyato, T. Kataoka, and M. Koyama, "Spectral normalization for generative adversarial networks," 2018, https://arxiv.org/abs/1802.05957.

[34] S. C. Martin Arjovsky and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the 34 th International Conference on Machine Learning*, Sydney, Australia, August 2017.

[35] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proceedings of the International Conference on Machine Learning*, Atlanta, GA, USA, June 2013.

[36] Y. Yoshida and T. Miyato, "Spectral norm regularization for improving the generalizability of deep learning," in *Proceedings of the International Conference on Neural Information Processing Systems*, pp. 1539–1550, Montreal, Canada, December 2018.

[37] H. Sedghi, V. Gupta, and P. M. Long, "The singular values of convolutional layers," 2018, https://arxiv.org/abs/1805.10408.

[38] I. Gulrajani, F. Ahmed, and M. Arjovsky, "Improved training of wasserstein gans," in *Proceedings of the Advances in neural Information Processing Systems*, pp. 5767–5777, Long Beach, CA, USA, December 2017.

[39] M. Abadi, P. Barham, and J. Chen, "Tensorflow: a system for large-scale machine learning," in *Proceedings of the 12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pp. 265–283, Carlsbad, CA, USA, July 2016.

[40] M. D. Zeiler, "Adadelta: an adaptive learning rate method," 2012, https://arxiv.org/abs/1212.5701.

[41] C. O. Ancuti, C. Ancuti, R. Timofte, and C. De Vleeschouwer, "I-HAZE: a dehazing benchmark with real hazy and haze-free indoor images," 2018, https://arxiv.org/abs/1804.05091.

[42] C. O. Ancuti, C. Ancuti, and R. Timofte, "O-haze: a dehazing benchmark with real hazy and haze-free outdoor images," in *in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 754–762, IEEE, Salt Lake City, UT, USA, June 2018.

[43] B. Li, W. Ren, and D. Fu, "Benchmarking single-image dehazing and beyond," *IEEE Transactions on Image Processing*, vol. 28, no. 1, pp. 492–505, 2018.

Hindawi
Complexity
Volume 2021, Article ID 2678124, 8 pages
https://doi.org/10.1155/2021/2678124

Research Article

MSLp: Deep Superresolution for Meteorological Satellite Image

Liling Zhao,[1,2] **Hao Yu**,[1] **and Yan Wang**[1]

[1]*School of Automation, Nanjing University of Information Science & Technology, Nanjing 210044, China*
[2]*School of Computer Science and Technology, Nanjing University of Science and Technology, Nanjing, China*

Correspondence should be addressed to Liling Zhao; zhaoliling@nuist.edu.cn

Received 30 July 2020; Revised 7 November 2020; Accepted 19 December 2020; Published 7 January 2021

Academic Editor: Zhijie Wang

High-resolution meteorological satellite image is the basic data for weather forecasting, climate prediction, and early warning of various meteorological disasters. However, the poor image resolution is limited for both subjective and automated analyses. Through our investigation and study, we found that the meteorological satellite image is a kind of complex data with multimodal and multitemporal characteristics. Fortunately, based on zero-shot learning theory, the complexity of the meteorological satellite image can be used to enhance its own image resolution. In this work, we propose a novel framework called MSLp (Meteorological Satellite Loss phase). Specifically, we choose a zero-shot network as a backbone and propose a phase loss function. A mapping from low- to high-resolution meteorological satellite images was trained for improving the resolution by up to a factor of 4×. Our quantitative study demonstrates the superiority of the proposed approach over ZSSR and bicubic interpolation. For qualitative analysis, visual tests were performed by 7 meteorologists to confirm the utility of the proposed algorithm. The mean opinion score is 9.32 (the full score is 10). These meteorologists think that weather forecasters need higher-resolution meteorological satellite images and the high-resolution images obtained by our method have the potential to be a great help for weather analysis and forecasting.

1. Introduction

Meteorological satellite images can objectively reflect the information of atmospheric structure, occurrence, and development around the Earth's surface [1]. Therefore, a high-resolution meteorological satellite image is an important basic dataset which is needed by the atmospheric science and meteorological industry. It plays a main role in the weather forecast, climate prediction, and early warning and defense of various meteorological disasters. However, due to the limitation of meteorological satellite optical system, manufacturing cost, and technological level, the spatial resolution of meteorological satellite images is not satisfied with applications [2] because it is difficult to find a small-scale atmospheric system or monitor the states and changes of a microscale atmosphere (such as air pollution sources and haze) in these low-resolution images.

Signal image superresolution (SISR) is a hot topic with great achievements in computer vision. It is defined as recovering a high-resolution (HR) image from its low-resolution (LR) observation [3]. This is a new method for high-resolution meteorological satellite images required by atmospheric science research. Furthermore, it can save the manufacturing cost of updating the meteorological satellite. However, since there are multiple solutions for any LR input, SISR is an ill-posed problem. In recent years, a variety of deep learning (DL) methods have been proposed to tackle such an inverse problem by learning mappings between LR and HR image pairs [4], such as SRCNN [5], VDSR [6], LapSRN [7], SRGAN [8], EDSR [9,10], and RCAN [11]. However, it does not work well by using these current methods directly on the problem of meteorological satellite image superresolution. The primary reason is the lack of many meteorological satellite image pairs. Therefore, meteorological satellite image superresolution is more challenging than the general SISR.

Inspired by the recent zero-shot learning theory, an unsupervised deep network needing no training data is designed in this paper. The proposed network overview is

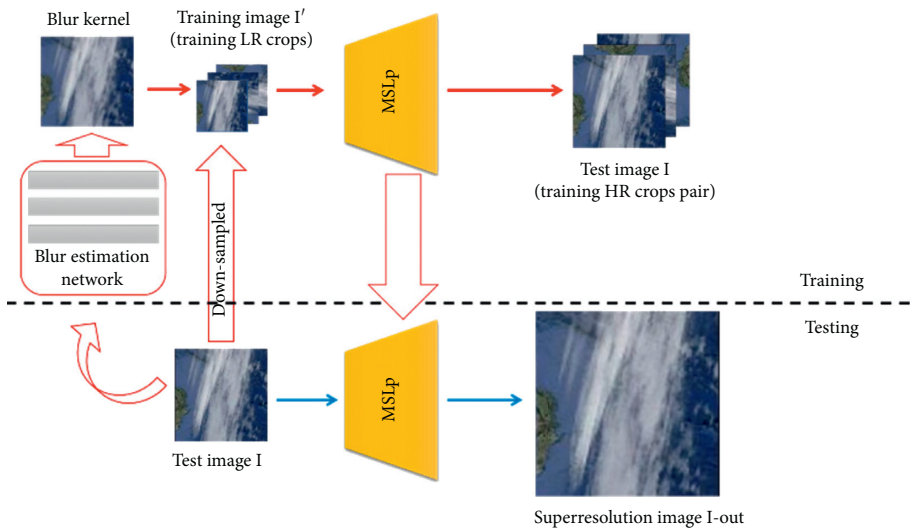


Figure 1: System overview. A proposed small CNN is trained on examples extracted internally from the test image itself. This CNN learns to recover the test image I from its low resolutions with the estimated blur kernel. The resulting unsupervised network is the applied LR image I to predict its HR.

shown in Figure 1. Our SISR methods can be realized with the internal features of the LR satellite image only. Moreover, we also propose a loss function based on the phase consistency principle. The phase loss function can monitor the network training and make the network robust for external influences such as scale change and complex degradation process. To the best of our knowledge, it is the first time to use phase for SISR CNN training. Experiment results show that guiding together with our phase loss function, the neural network for meteorological satellite image superresolution can achieve better performance. The main contributions of this paper are listed as follows:

Zero-shot learning-based superresolution network: we propose a zero-shot learning-based superresolution unsupervised deep neural network architecture for meteorological satellite image superresolution task.

Phase loss function: we introduce phase loss which is a loss function specifically optimized for satellite image images. It can measure the difference between generated HR images and ground truth in the frequency domain. Combining this phase loss function with other spatial losses can improve image quality in terms of pixel values and texture.

In the rest of this paper, we review the related works of unsupervised deep network and loss functions in Section 2. The network architecture and the phase loss function are discussed in Section 3. In Section 4, the results of both quantitative and qualitative evaluations are presented. Finally, Section 6 concludes this paper.

2. Related Work

2.1. Unsupervised Superresolution. Most of the existing superresolution works are supervised learning, i.e., learning the LR-to-HR mapping using matched LR-HR image pairs. However, since it is difficult to collect images of the same scene of different resolutions and the HR images are not even available, the SR models trained on these datasets are more likely to learn a reverse version of the predefined process. In order to prevent the adverse effects brought by predefined degradation, researchers pay more attention to the unsupervised superresolution method. Now, the SISR models can be used in reconstructing the degraded images in the real world. Next, we will briefly introduce several existing unsupervised SR models based on deep learning.

Considering that the structure of a CNN is sufficient to capture a lot of low-level image statistics prior to inverse problems, Ulyanov et al. [12] employ a randomly initialized CNN as handcrafted prior to perform SR. Because the network is randomly initialized and never trained on datasets, the only prior is the CNN structure itself. It shows the rationality of the CNN architectures itself and prompts us to improve superresolution by combining the deep learning methodology with handcrafted priors such as CNN structures or self-similarity. Another approach for unsupervised superresolution is to treat the LR space and the HR space as two domains and use a cycle-in-cycle structure to learn the mappings between each other. Motivated by CycleGAN [13], Yuan et al. [14] propose a cycle-in-cycle SR network (CinCGAN) composed of 4 generators and 2 discriminators. Because of avoiding the predefined degradation, the unsupervised CinCGAN not only achieves comparable performance to supervised methods but also is applicable to various cases even under very harsh conditions. In 2018, considering that the internal image statistics inside a single image is sufficient to provide the information needed for superresolution, ZSSR [15] used LR sons that are downsampled images of the given LR test image. Using this pseudopairs, they train small image-specific SR networks at test time rather than training a generic model on large external datasets. Specifically, they use a kernel estimation method [16] to directly estimate the degradation kernel from a single test image and use this kernel to construct a small

dataset by performing degradation with different scaling factors on the test image. Then, a small CNN for superresolution is trained on this dataset and used for the final prediction.

In the application of meteorological satellite image processing, real-world images tend to be suffering from unknown degradation, for example, the additive noise, compression artefacts, and blurring, etc. We cannot get the exact mathematical model. To perform this unsupervised superresolution method for meteorological satellite images, some advanced strategies like architecture, active function, and loss function are also needed for reducing the training difficulty and instability.

2.2. ZSSR and Loss Function. In our research, we choose ZSSR as the backbone. ZSSR uses the deep neural network architecture for superresolving the images only based on the internal image statistics. The aim is to predict the test image from the LR image created from the test image. Hence, it does not require a training image as an input. The ZSSR has eight convolutional layers followed by ReLU consisting of 64 channels and learns the residue image using + loss. Besides a good architecture of ZSSR, robust learning strategies are also needed for achieving satisfactory results. Next, we will discuss the loss function which is a promising direction of learning strategies.

In the early researches, l_1 and l_2 are usually employed as the most widely used loss function for network optimization. But latterly, it has been discovered that these pixelwise loss functions cannot take image quality into account; they often lack high-frequency details and produce perceptually unsatisfying results with overly smooth textures [17–19]. Therefore, a variety of loss functions are adopted to better measure the reconstruction error. To evaluate image quality based on the perceptual, the content loss is introduced [20]. In contrast to the pixel loss, the content loss encourages the output image to be perceptually the same as the target image instead of forcing them to match pixels exactly. Specifically, it measures the semantic differences between images using a pretrained image classification network. Thus, it produces visually more perceptible results and is widely used for SISR [21, 22]. On account that the reconstructed image should have the same style (e.g., colors, textures, and contrast) as the target image, the texture loss is introduced into superresolution, based on the style representation motivation in [23, 24]. By employing texture loss, the SR model can create realistic textures and produce visually more satisfactory results. In recent years, the GANs [25] have been introduced to various computer vision tasks. To be concrete, especially in the superresolution field, Ledig et al. [20] firstly introduce SRGAN by using adversarial loss based on cross-entropy. As a matter of fact, the discriminator extracts some latent patterns of real HR images and conforms the generated HR images; this helps to generate more realistic images [26, 27]. In 2017, Yuan et al. [14] present a cycle consistency loss for image superresolution. Concretely speaking, the cycle-in-cycle approach motivated by CycleGAN [13] superresolves the LR image to the HR image by constraining image pixel-level consistency. In order to suppress noise in generated images, the total variation (TV) loss is introduced by Aly et al. [28]. It is defined as the sum of the absolute differences between neighbouring pixels and measures how much noise is in the images. In addition to the above loss functions, external prior knowledge is proposed to constrain the generation process [27]. By introducing more prior knowledge, the performance of superresolution can be further improved.

In general, different loss functions can measure one kind of errors specifically. In practical application, in order to get better superresolution results, more than one of the loss functions are combined to calculate errors. However, from the l_1 loss to the external prior loss, all these loss functions are designed in the spatial domain. No matter how to make the combinations, it is only an error measurement of spatial features. Different from these methods, we will discuss a new phase loss function in the frequency domain, which can minimize the phase difference between the generated HR image and ground truth. More details will be shown in Section 3.

3. Methods

Considering the training dataset is insufficient for meteorological satellite image superresolution task. We propose a CNN network based on zero-shot learning theory and give a new phase loss to monitor this unsupervised network training. In the following subsections, we first introduce the proposed CNN structure in Section 3.1. Then, we discuss the phase loss functions designed in our approach in Section 3.2.

3.1. Proposed CNN Architecture. Following the methods of Wang [29], we propose a neural network for meteorological satellite image superresolution. The proposed network aims to estimate the SR meteorological satellite image from its LR image only, synthesizing plausible textures while preserving the consistency with LR in content. The architecture of the proposed MSLp network inspired by the paper [15] is shown in Figure 2. Considering that the internal image statistics inside a single meteorological satellite image are sufficient to provide the information needed for superresolution, we cope with unsupervised SR by training small image-specific SR networks at test time rather than training a generic model on large external datasets. The proposed network model consists of 8 hidden layers, each has 64 channels with the activation being ReLU except the last layer.

3.2. Loss Function. Our goal is to find the sharp image from a single low-resolution meteorological satellite image. The degeneration image can be modelled as a convolution on the sharp image with a subsampled kernel.

$$I_L = S \otimes I_H, \tag{1}$$

where I_H is the known HR image, I_L denotes the LR image, S is the subsampled kernel, and $\otimes$ is the convolution operator. This is a highly ill-posed problem, since multiple solutions for any I_H and S can lead to the same I_L. In the Fourier

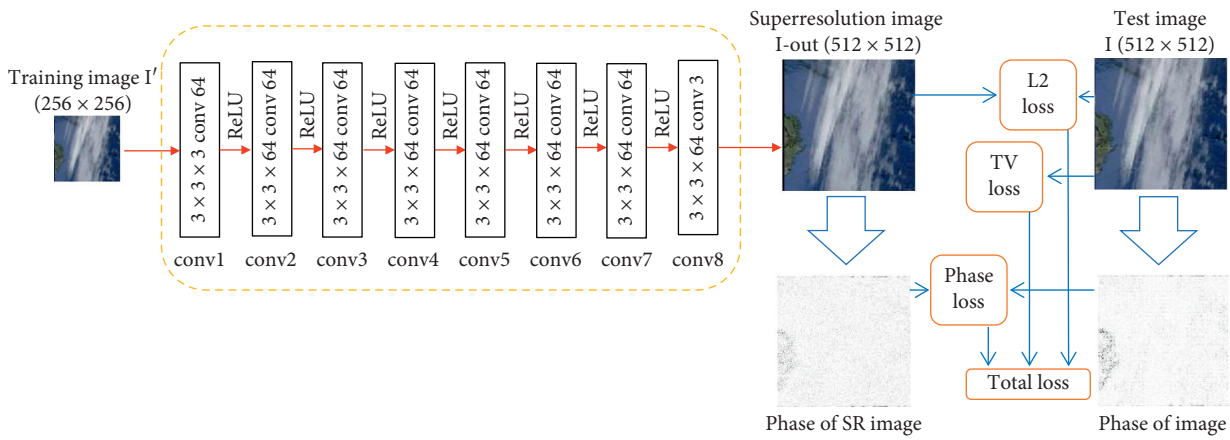


Figure 2: The architecture of MSLp superresolution framework. On the left, a low-resolution image obtained by its real image with the estimated blur kernel is fed to the network to produce a superresolution image, which is then passed to the total loss function combined with spatial and frequency loss to measure the difference between the generated image and the target image.

domain, equation (1) corresponds to $F(I_L) = F(S) \odot F(I_H)$, where $\odot$ represents the component-wise multiplication and F is the Fourier transform. The phase and amplitude of a complex number $z = ke^{i\theta}$ are $e^{i\theta}$ and $k \geq 0$, respectively. We denote taking the phase of a complex signal by $\mathscr{P}(\cdot)$. Taking the inverse Fourier transform F^{-1} of the phase component gives the phase-only image, which can be formulated as

$$P(I_H) = F^{-1}(\mathscr{P}(F(I_H))). \tag{2}$$

From the previous analysis, we know that the phase recovering is certainly helpful for SISR. Therefore, we consider that in order to make full use of the advantages of spatial and frequency domain information, we propose to jointly model the loss function in the spatial domain and frequency domain. However, how to recover the real phase in the frequency domain is related to the fact, noted in [30], that the Fourier components of an edge tend to be in-phase with each other. In this paper, we define a loss function called phase loss to measure the difference between the superresolution image generated by the deep learning network and the real high-resolution image.

The definition of our phase loss l_{ph} is critical for the performance of our neural network. The phase loss is calculated as

$$l_{\text{Ph}}^{\text{SR}} = \frac{1}{rWH} \sum_{i=1}^{rW} \sum_{j=1}^{rH} \left(P\left(I_{i,j}^{\text{HR}}\right) - P\left(I_{i,j}^{\text{LR}}\right)\right)^2, \tag{3}$$

where $l_{\text{Ph}}^{\text{SR}}$ is the phase loss of SR image, r is the downsampling factor, W and H are the width and height of image, $I_{i,j}^{\text{HR}}$ is the HR image, $I_{i,j}^{\text{LR}}$ is the LR image, and $P(I_{i,j}^{\text{HR}})$ and $P(I_{i,j}^{\text{LR}})$ are the phase-only image which is described by equation (2), respectively.

In this paper, in addition to the phase loss function described so far, we also use spatial loss, such as pixel loss and TV loss in our MSLp network. Therefore, the loss function of our work is

$$l = \lambda_1 l_{\text{spa}} + \lambda_2 l_{\text{ph}}, \tag{4}$$

where l_{spa} is the spatial loss and l_{Ph} is the phase loss. λ_1, λ_2 are the coefficients to balance different loss terms. In our experiment, l_{spa} will be replaced by l_2 and l_{TV} combinations. In our work, l_2 is the pixel loss in the spatial domain calculated as

$$l_2 = \frac{1}{rWH} \sum_{i=1}^{rW} \sum_{j=1}^{rH} \left(I_{i,j}^{\text{HR}} - I_{i,j}^{\text{LR}}\right)^2, \tag{5}$$

l_{TV} is the total variation loss formulated by

$$l_{\text{TV}} = \frac{1}{rWH} \sum_{i=1}^{rW} \sum_{j=1}^{rH} \left(V_{i,j}^{\text{HR}} - V_{i,j}^{\text{LR}}\right)^2, \tag{6}$$

and V is the total variation; the form is $V = \sum_{i,j} |I_{i+1,j} - I_{i,j}| + |I_{i,j+1} - I_{i,j}|$.

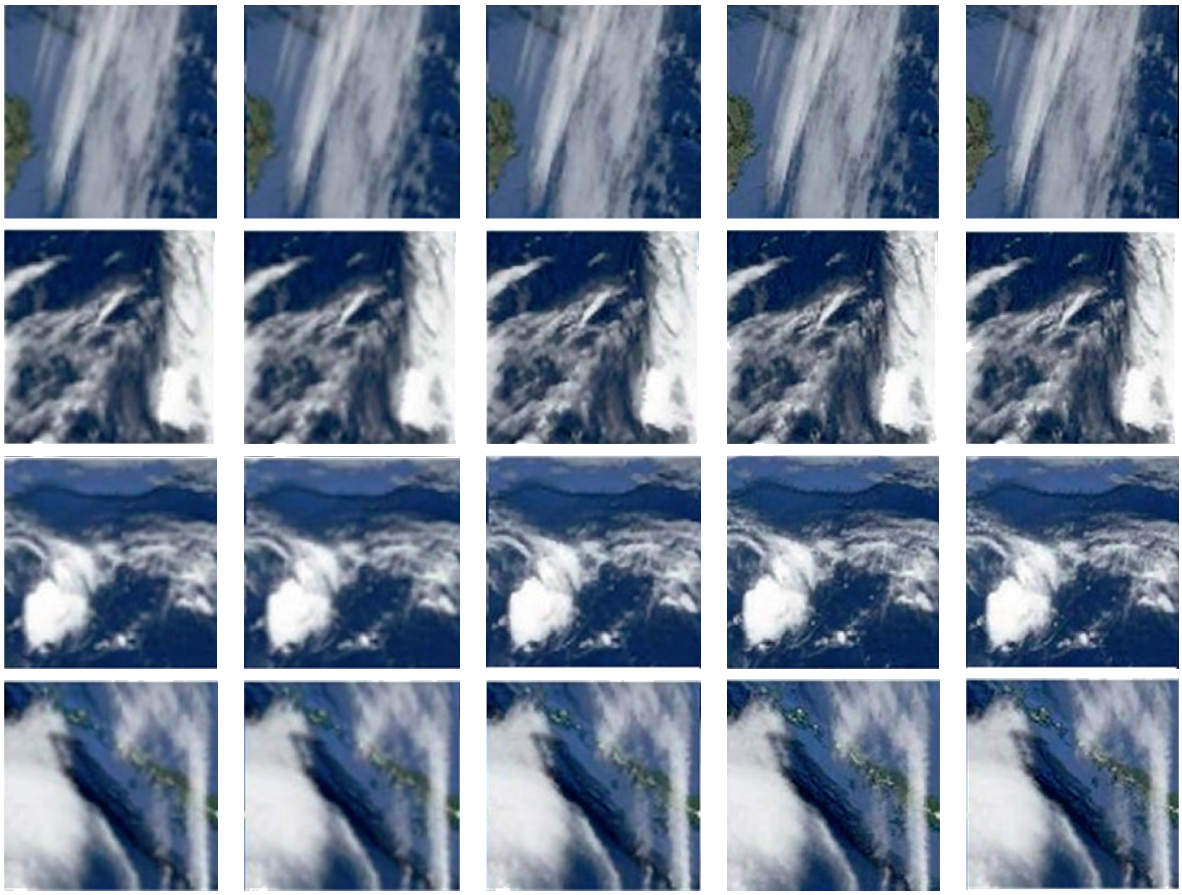
In our experiments, we will discuss the effect of this phase loss and other spatial loss functions in meteorological satellite image superresolution. Moreover, we will make a mechanism study based on our proposed network to further explain the phase loss effectiveness and give the best loss combination in the spatial and frequency domain.

4. Experiments

4.1. Data Preparation. We chose a portion of the NSMC (National Satellite Meteorological Centre) dataset for training and testing. This dataset provides visible and infrared images every hour, and the portion we used consists of 500 images of it. In our experiments, we first use a kernel estimation method [16] to directly estimate the degradation kernel from a single test image and use this kernel to construct a small "training set" by performing degradation with different scaling factors on the test image. Since the above "training set" consists of one instance only (the test image), we employ data augmentation on the test image to extract more LR-HR example pairs. The augmentation is done by downscaling the test image to many smaller versions

Table 1: Mean PSNR and SSIM of ablation experiment (500 testing images).

	l_2	$l_{2+3\times10^{-3}l_{\mathrm{Ph}}}$	l_{TV}	$l_{\mathrm{TV}+3\times10^{-3}l_{\mathrm{Ph}}}$	$l_{2+2\times10^{-6}l_{\mathrm{TV}}+2\times10^{-6}l_{\mathrm{Ph}}}$
PSNR	34.72	35.15	31.78	35.20	35.92
SSIM	0.84	0.84	0.71	0.82	0.89



(a) (b) (c) (d) (e)

Figure 3: Comparison of superresolution results for visible images (4×). (a) Degraded image. (b) Bicubic. (c) ZSSR. (d) MSLp. (e) Real image.

of itself. These play the role of HR supervision and are called "HR fathers." Each of the HR fathers is then downscaled by the desired SR scale factor to obtain the "LR sons," which form the input training instances. The resulting training set consists of many image-specific LR-HR example pairs. Our network can then stochastically train over these pairs.

4.2. Investigation of Phase Loss. Several spatial loss types have been investigated to guide the MSLp network convergence. The L1-norm pixel loss, the TV loss, and the phase loss are the main components of the loss function as described in the previous sections. The overall loss function of our network is

$$L_{\mathrm{MSLp}} = \alpha l_2 + \beta l_{\mathrm{TV}} + \gamma l_{\mathrm{ph}}. \tag{7}$$

l_2 is the pixel loss between the target HR image and the SR image in the spatial domain described in equation (5). l_{TV} represents the gradient amplitude summation of the SR image, described in equation (6). The variable α dynamically modulates the influence of the pixel loss on the overall loss. Similarly, β controls the noise to keep the SR image smooth. The variable γ controls the importance of the phase loss to evaluate whether the recovery of high-frequency information is correct. We investigated the effect of the loss choices for the MSLp network. Quantitative results are listed in Table 1. Adding l_{Ph} to spatial loss can raise the PSNR and SSIM value while combining $l_2 + l_{\mathrm{TV}} + l_{\mathrm{ph}}$ together will give better results. It shows that joining the frequency loss into spatial loss will get a beneficial result when training an unsupervised super-resolution neural network. In our experiments, the variables α, β, and γ can be adjusted. In Table 1, we give the best combinations when standard quantitative measures such as PSNR and SSIM reach the highest measures.

4.3. Qualitative Evaluation. We trained and tested our MSLp network on TITAN X NIVIDA GPU and Intel (R) Core (TM) i7-4790 3.6 GHz with 16 GB DDR3 1600 RAM.

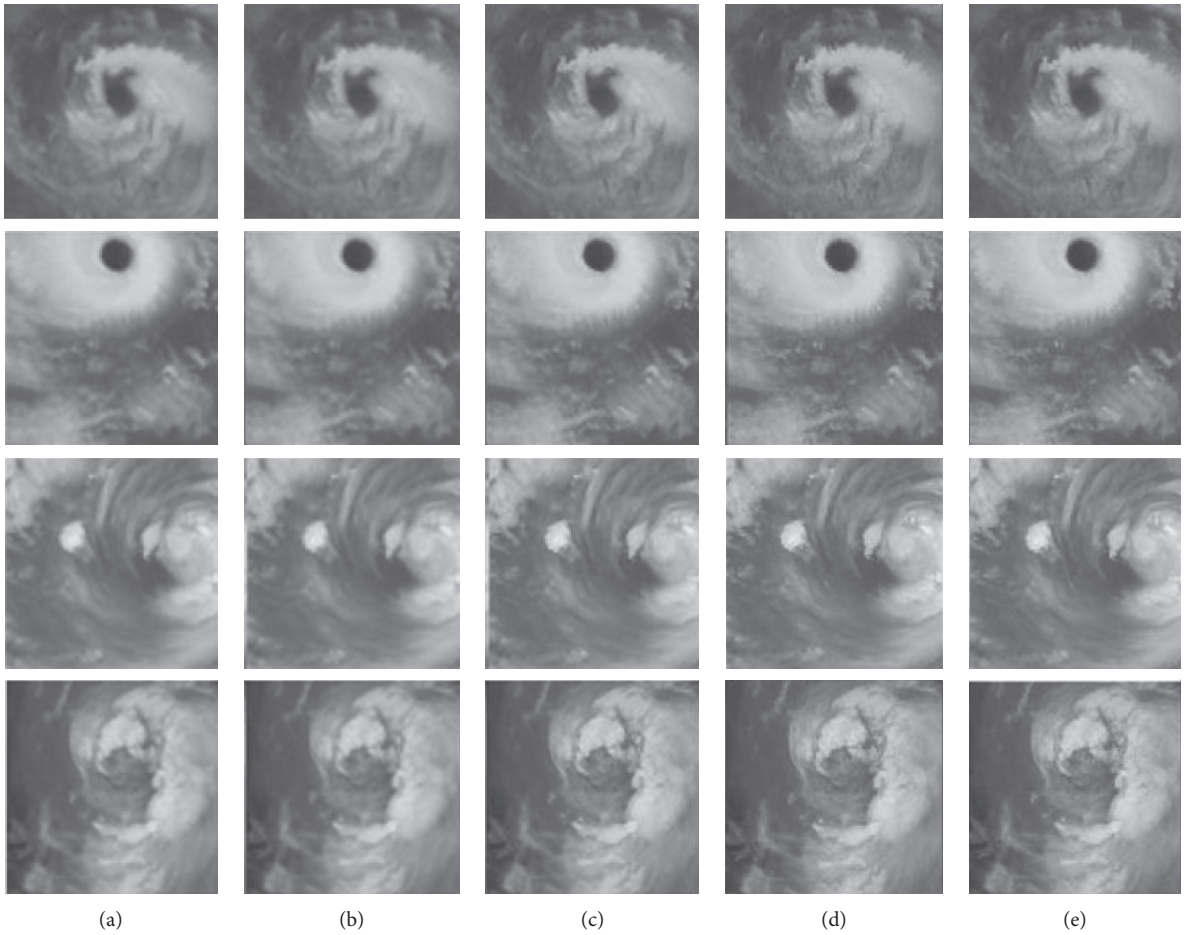


Figure 4: Comparison of superresolution results for infrared images (4×). (a) Degraded image. (b) Bicubic. (c) ZSSR. (d) MSLp. (e) Real image.

The net models are trained using Pytorch 0.4.1. The network input is interpolated to the output size and blurred with the estimated degradation kernel. As done in previous ZSSR-based SR methods, we only learn the residual between the LR and its HR pairs. We use L_{MSLp} loss with Adam optimizer. We start with a learning rate of 0.001. We periodically take a linear fit of the reconstruction error and we change the learning rate by 0.05~1.5.

4.3.1. Objective Evaluations. Peak signal-to-noise ratio (PSNR) is a full-reference image quality measure of pixelwise similarity. Higher PSNR values represent greater pixelwise similarity. It is a widely used image quality metric, and it should be considered when comparing performances with other models. Structural similarity index (SSIM) is a well-known full-reference quality metric that takes into account the structural composition of pixels. By making use of contrast, luminance, and structure values of images, SSIM aims to measure perceived quality by the human visual system, and it gets values between 0 and 1. As it can be seen from Table 1, the MSLp network ($l_2 + l_{\mathrm{TV}} + l_{\mathrm{ph}}$) overperforms ZSSR and bicubic interpolation in terms of PSNR and SSIM. We compared the performance of the MSLp with the state-of-the-art ZSSR and bicubic interpolation approaches. Figure 3 shows the SR results of the meteorological satellite visible images. The row is, from left to right, degraded images estimated from the "blur estimation network," bicubic, ZSSR, MSLp, and the ground truth. Figure 4 shows the SR results of the meteorological satellite infrared images.

Table 2: Mean opinion score results.

	Detail level			Sharpness		
	MSLp	ZSSR	Bicubic	MSLp	ZSSR	Bicubic
Mean	9.32	8.32	6.76	9.31	8.35	7.52
Max	9.78	8.90	7.12	9.70	8.90	8.21
Min	8.72	7.55	6.13	8.93	7.57	6.89
Std	0.39	0.45	0.34	0.23	0.46	0.38

4.3.2. Mean Opinion Score Evaluations. Mean opinion score is a survey-based subjective image quality metric in which humans participants are asked to choose a score regarding a set of specified features (e.g., sharpness). Some SR models might have low PSNR and SSIM scores whereas the perceptual quality of HR images could be still high, in which case MOS testing should be employed. Specifically, we asked

7 meteorologists to assign an integral score to 10 output images from 1 (bad quality) to 10 (excellent quality) for each of 3 comparison metrics and reconstructed images, respectively. MOS results for all reference methods on the realistic meteorological satellite image are listed in Table 2. We performed a MOS test to subjectively quantify the sharpness, suitability for weather analysis, and detail level of the reconstructed superresolution satellite images of different approaches. In these two evaluation indicators, detail level and sharpness, the mean of MSLp is maximum while the Std of MSLp is minimum. This shows that most meteorologists are more satisfied with our results, and there is little deviation between them. The new method results are more acceptable than all the reference methods.

5. Conclusions

We have introduced an unsupervised deep superresolution network MSLp specifically designed and optimized for meteorological satellite images. The proposed network combines spatial and frequency loss functions by network training. Our investigation on network structure indicates that the proposed unsupervised deeper networks are more beneficial in meteorological satellite image superresolution tasks. We demonstrate the performance of MSLp using quantitative and qualitative tests for (4×) upscaling factor and compared it with the state-of-the-art reference methods. The test results prove that reconstructions of MSLp are perceptually more plausible in terms of weather analysis applications. Last but not least, our method showed promising results in meteorological satellite image unsupervised superresolution, implying its potential extensibility to other unsupervised low-level vision tasks.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request. The other supplementation used to support the findings of this study are supplied by all the authors under license and so cannot be made freely available. Requests for access to these data should be made to the corresponding author.

Conflicts of Interest

All authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (grant number 61802199).

References

[1] T. Fan, *Meteorological Satellites and Satellite Meteorology*, pp. 101–105, China Meteorological Press, Beijing, China, 2014.

[2] R. Huang, *Major Scientific Issues in Atmospheric Science and Global Climate Change Research*, Science Press, Beijing, China, 2016.

[3] C. Y. Yang, C. Ma, and M. H. Yang, "Single-image super-resolution: a benchmark," in *Proceedings of the European Conference on Computer Vision*, Springer International Publishing, Zurich, Switzerland, September 2014.

[4] Z. Chu, *Research on Super Resolution Reconstruction Algorithms*, Nanjing University, Nanjing, China, 2016.

[5] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Proceedings of the European Conference on Computer Vision*, Springer, Zurich, Switzerland, September 2014.

[6] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate image super resolution using very deep convolutional networks," in *Proceedings of the Computer Vision and Pattern Recognition CVPR*, Las Vegas, NV, USA, June 2016.

[7] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Deep Laplacian pyramid networks for fast and accurate super resolution," in *Proceedings of the Computer Vision and Pattern Recognition CVPR*, Honolulu, HI, USA, July 2017.

[8] C. Ledig, L. Theis, F. Huszar et al., "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4681–4690, IEEE Computer Society, Honolulu, Hawaii, USA, July 2017.

[9] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image superresolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops CVPRW*, Honolulu, HI, USA, July 2017.

[10] Y. Wang, F. Perazzi, B. McWilliams, A. Sorkine-Hornung, O. Sorkine-Hornung, and C. Schroers, "A fully progressive approach to single-image superresolution," in *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops*, Salt Lake City, UT, USA, June 2018.

[11] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super resolution using very deep residual channel attention networks," in *Proceedings of the European Conference on Computer Vision*, pp. 294–310, Springer, Munich, Germany, September 2018.

[12] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proceedings of the Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, June 2018.

[13] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the International Conference on Computer Vision ICCV*, Venice, Italy, October 2017.

[14] Y. Yuan, S. Liu, J. Zhang, Y. Zhang, C. Dong, and L. Lin, "Unsupervised image superresolution using cycle-in-cycle generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops CVPRW*, Salt Lake City, UT, USA, June 2018.

[15] A. Shocher, N. Cohen, and M. Irani, "Zero-shot super-resolution using deep internal learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition CVPR*, Salt Lake City, UT, USA, June 2018.

[16] T. Michaeli and M. Irani, "Nonparametric blind super-resolution," in *Proceedings of the IEEE International Conference on Computer Vision ICCV*, Sydney, Australia, December 2013.

[17] Z. Wang, E. Simoncelli, and A. Bovik, "Multi-scale structural similarity for image quality assessment," in *Proceedings of the Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, USA, November 2003.

[18] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural

similarity," *IEEE Transactions on Image Processing*, vol. 13, 2004.

[19] M. S. Sajjadi, B. Sch¨olkopf, and M. Hirsch, "Enhancenet: single image superresolution through automated texture synthesis," in *Proceedings of the IEEE International Conference on Computer Vision ICCV*, Venice, Italy, October 2017.

[20] C. Ledig, L. Theis, F. Husz´ar et al., "Photorealistic single image superresolution using a generative adversarial network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, July 2017.

[21] X. Wang, K. Yu, C. Dong, and C. C. Loy, "Recovering realistic texture in image super-resolution by deep feature transform," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, June 2018.

[22] X. Wang, K. Yu, S. Wu et al., "Enhanced superresolution generative adversarial networks," in *Proceedings of the ECCV Workshop*, Munich, Germany, September 2018.

[23] L. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," in *Proceedings of the Conference on Neural Information Processing Systems NIPS*, Montreal, Canada, December 2015.

[24] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proceedings of the Conference on Computer Vision and Pattern Recognition CVPR*, Las Vegas, NV, USA, June 2016.

[25] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., "Generative adversarial nets," in *Proceedings of the Conference on Neural Information Processing Systems NIPS*, Montreal, Canada, December 2014.

[26] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real time style transfer and superresolution," in *Proceedings of the European Conference on Computer Vision ECCV*, Amsterdam, The Netherlands, October 2016.

[27] A. Bulat and G. Tzimiropoulos, "Super-fan: integrated facial landmark localization and superresolution of real-world low resolution faces in arbitrary poses with GANs," in *Proceedings of the Conference on Computer Vision and Pattern Recognition CVPR*, Salt Lake City, UT, USA, June 2018.

[28] H. A. Aly and E. Dubois, "Image up-sampling using total variation regularization with a new observation model," *IEEE Transactions on Image Processing*, vol. 14, 2005.

[29] Z. Wang, J. Chen, and S. C. H. Hoi, "Deep learning for image super-resolution: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, .

[30] P. Kovesi, "Phase congruency detects corners and edges," in *Proceedings of the Conference on DICTA*, Sydney, Australia, December 2003.

Hindawi
Complexity
Volume 2020, Article ID 6629634, 8 pages
https://doi.org/10.1155/2020/6629634

Research Article

Realistic Speech-Driven Talking Video Generation with Personalized Pose

Xu Zhang and Liguo Weng

Jiangsu Key Laboratory of Big Data Analysis Technology, Nanjing University of Information Science and Technology, Nanjing 210044, China

Correspondence should be addressed to Liguo Weng; liguoweng@hotmail.com

Received 30 October 2020; Revised 18 November 2020; Accepted 8 December 2020; Published 29 December 2020

Academic Editor: Zhijie Wang

In this work, we propose a method to transform a speaker's speech information into a target character's talking video; the method could make the mouth shape synchronization, expression, and body posture more realistic in the synthesized speaker video. This is a challenging task because changes of mouth shape and posture are coupled with audio semantic information. The model training is difficult to converge, and the model effect is unstable in complex scenes. Existing speech-driven speaker methods cannot solve this problem well. The method proposed in this paper first generates the sequence of key points of the speaker's face and body postures from the audio signal in real time and then visualizes these key points as a series of two-dimensional skeleton images. Subsequently, we generate the final real speaker video through the video generation network. We take a random sampling of audio clips, encode audio contents and temporal correlations using a more effective network structure, and optimize and iterate network outputs using differential loss and attitude perception loss, so as to obtain a smoother pose key-point sequence and better performance. In addition, by inserting a specified action frame into the synthesized human pose sequence window, action poses of the synthesized speaker are enriched, making the synthesis effect more realistic and natural. Then, the final speaker video is generated by the obtained gesture key points through the video generation network. In order to generate realistic and high-resolution pose detail videos, we insert a local attention mechanism into the key point network of the generated pose sequence and give higher attention to the local details of the characters through spatial weight masks. In order to verify the effectiveness of the proposed method, we used the objective evaluation index NME and user subjective evaluation methods, respectively. Experiment results showed that our method could vividly use audio contentsto generate corresponding speaker videos, and its lip-matching accuracy and expression postures are better than those of previous work. Compared with existing methods in the NME index and user subjective evaluation, our method showed better results.

1. Introduction

The task of a speech-driven speaker video refers to a technology that automatically generates a video of a corresponding character's speech through a computer-based audio information. The content of the talking must be consistent with the character's pose in the video. Traditional speech-driven talking video requires professional equipments and operators to perform character modeling, which is usually very expensive for custom use. In recent years, with the successful application of deep neural networks, data-driven speech and video synthesis methods have been proposed. These methods often require the use of a large amount of high-quality audio and video data, and the production process is complex, but the synthesized speaker's mouth posture matching effect is poor.

The current mainstream methods mainly focus on facial speaker synthesis and do less work on body postures and facial expressions. Specifically, the existing methods [1, 2] input the speaker's voice information into the recurrent neural network to obtain 3D face model parameters, then map the fitted 3D face model to 2D key points as inputs of the video synthesis module, and then output corresponding speaker pictures through the video synthesis model. Due to

the weak representation ability of the 3D face model parameter network, the key point error obtained from the 3D face model conversion is larger, the 3D face model needs to be used as an intermediate state for conversion, resulting in a complicated overall process. Eskimez et al. [3] converted the facial key points into the average face space in the dataset to remove ID features and simplified the task. Although the key point indicators obtained from the network output are relatively low, the posture expressionsare very monotonous and rigid, and hence , the synthesized speaker video is not realistic enough.

As mentioned above, the matching effect of existing speech-driven speaker methods is not ideal, and the synthesized speaker video has a jitter phenomenon. In order to solve the above problems, this paper proposes a method to convert the speaker's voice information into the target person's talking video. We use the Dilated Depthwise Separable Residual (DDSR) unit to encode the audio features [4, 5], and then use the GRU network layer [6] to learn the temporal features and constrain the network outputs using content loss functions. Through this network structure, the audio content and temporal correlation information are effectively encoded simultaneously, the facial key point index of the model output is lowered, and the mouth shapes and postures of the synthesized speaker video are matched with audio contents better, plus, the synthesized speaker video is more natural and realistic. In the process of training and testing, we insert the specified pose sequence frame into the pose sequence, which makes the audio conversion to the speaker's mouth shape and posture more natural and vivid. In order to enrich the speaker's detailed texture, we introduce a local attention mechanism in the key point network and add spatial weights to the face, fingers, and other parts of the character to get higher attentions.

Finally, in order to better evaluate our system, we used high-resolution and frame rate (FPS) cameras to create a dataset containing audio and video for multiple targets reading selected articles. Compared with the existing methods, our method produces better visual perception. In Figure 1, we show some images of our synthesized speaker video.

In summary, the contributions of our work are

(1) We use a novel Dilated Depthwise Separable Residual (DDSR) unit. This network structure can effectively represent the audio content and temporal correlation, and the facial key point index of the model output is lower. At the same time, the network model is used to model the key points of the face and human posture, respectively. After preprocessing , it uses the loss function to optimize iteratively. The results show that the face details and human postures are better.

(2) We use the first-order differential loss function and the pose perception loss function [7, 8] to optimize the model. Among them, the first-order differential loss function can smooth the pose of the front and rear frames, and the pose perception loss function uses the spatiotemporal graph to form a hierarchical representation of the pose sequence, so as to constrain the temporal-spatial information output from the network.

(3) We establish a pose keypoint map to add richer poses and expressions to the generated human poses. In addition, we also provide a method to convert the pose in the existing sequence window into the corresponding keyframe pose sequence.

2. Related Work

Given a speaker's audio information, the generation of the corresponding person speaking video has attracted many researchers' interests. Earlier works mainly used the Hidden Markov model (HMM) to generate corresponding relationships between speech and facial motions [9–14]. Among them, Brand [15] proposed voice puppetry as an HMM-based method for generating conversation faces driven only by voice signals. In another study, Cosker et al. [10, 11] proposed a hierarchical model that can animate the subregions of the face independently of speech and merge them into a complete face video.

In recent years, with successful applications of deep neural networks, the related work of speech-driven speaker based on deep learning method has been proposed. Among them, Suwajanakorn et al. [16] designed an LSTM network to directly generate the target identity talking face video from the audio. However, this method needs to record a large number of facial videos with specific target identities, it limits its application in many scenarios. Linsen et al. converted audio information into the 3D face model parameter space and then the fitted 3D face model to 2D facial key points. Their network uses several layers of recurrent neural networks as encoding, and the network feature learning ability is relatively weak. The facial key points obtained by the conversion of the 3D face model have a large error, and the 3D face model needs to be used as an intermediate state for conversion. This leads to the complexity of the overall process.

In addition, including the single-stage method of direct conversion of audio to speaker video space, many researchers divide the task of speech generation into two stages. Usually, the key point information only responds to the voice content information. Pham et al. [17] first used the LSTM network to map voice features to 3D deformable shapes and rotation parameters and finally generated 3D animated faces in real time based on the predicted parameters. In literature [18], they further improved this method, replacing speech features with original waveforms as inputs and the LSTM network with a convolutional structure. However, compared with the speech-generated gesture keypoint network in our method, their method is less intuitive in shapes and rotation parameters, and the mapping from these parameters to specific gestures or facial expressions is not clear. In another related work, the key points of the face that they generated are for a standardized average face, rather than for a specific target identity. Although this helps to eliminate factors that are not directly related to voice, the predicted sequence of key points for the

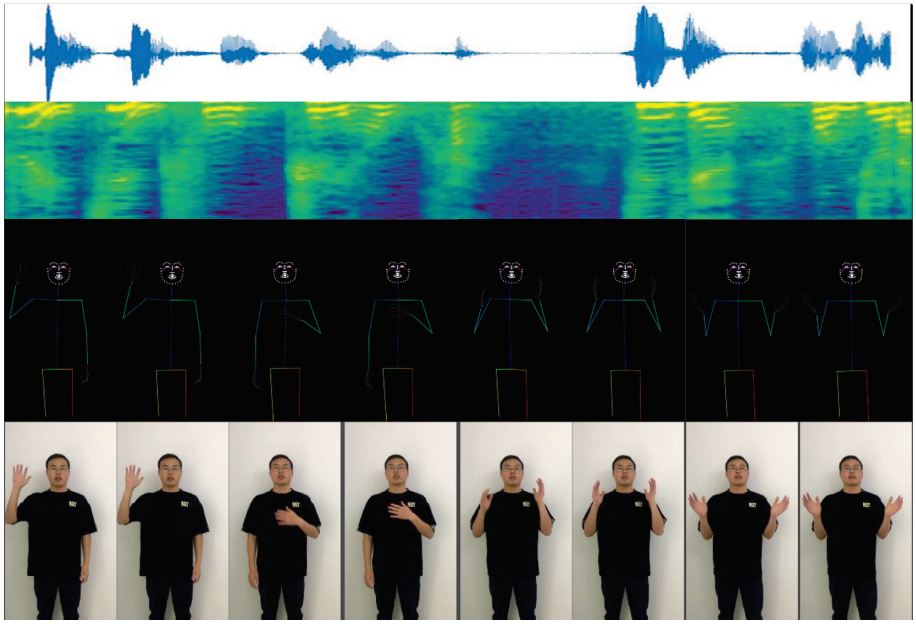


Figure 1: Speech-driven talking video: a given piece of audio/text can be used to drive the video of the specified speaker.

posture is unnatural. [19] An extended complex human motion synthesis method based on autotuning recurrent network is proposed. They can simulate more complex movements, including dances or martial arts. In the second stage of work, most methods use vid2vid [20] to enhance the time consistency between adjacent frames. Shysheya et al. [21] proposed a method to generate realistic videos from skeleton sequences without establishing a 3D model. Our method also uses the vid2vid network to synthesize the final speaker video from the posture skeleton picture and obtains better results. For the detailed texture information of the face and hands, we use separate discriminators to optimize these parts in vid2vid.

Our method expands the data of random audio samplings and uses a more effective network structure to learn audio contents and timing correlations. The loss function uses the first-order differential loss and poses perception loss to optimize output pose timing stability and matching accuracy. At the same time, the keyword wake-up technology is used to convert the generated sequence poses into specified action poses. A large number of experimental results show that our method generates a natural and realistic speaker video for talking audio, and its lip matching and expression posture are more expressive than those of the previous work.

3. Methods

In this section, we mainly introduce different modules of the network. The overall network structure is shown in Figure 2. In our approach, the input information can be either audio or text. When the audio information is used as the speaker synthesis network input, we convert the audio data into log-mel features; the aud2kps network is used to get the human body postures and facial key points. Using the Dictionary Building and Key Pose Insertion method to insert a specified action frame into the generated key point sequence, the synthesis effect is more natural and realistic, and then the output key points of facial and human posture are visualized as a series of 2D skeleton images, and these 2D skeleton images are further fed into the Vid2vid generation network to generate the final talking images. When the input is text information, it is necessary to use the acoustic model to convert the text information to obtain a unified log-mel feature as the input of the Aud2Kps network. The following steps are the same as the audio signal input process. The text-to-speech method (TTS) is currently very mature and commercialized, and we use the open source tactron2 [22] to complete the text conversion results which we want. In the following sections, we describe each module of our architecture.

3.1. Pose Keypoints. In the process of audio-video conversion, we use the key points of human body posture as the intermediate state representation so that the span of the two spatial features will not be too large. Compared with using the 3D human body model as the intermediate state representation, it is more convenient and universal in the process of training and reasoning. We use the open source method OpenPose [23, 24] to obtain the key points of the human body posture. These key points include a total of 137 position coordinate information of the body, feet, hands, and faces. Firstly, we construct these 2D key points and audio information into a content sequence and then train the Aud2Kps network to generate 2D coordinates corresponding to the posture key points from the audio speech information.

3.2. Audio to Keypoints (Aud2Kps). As shown in Figure 2, our Aud2Kps network takes log-mel spectrogram as the input. $[x_0, x_1, \ldots, x_n,]$ is the input vector of audio/text

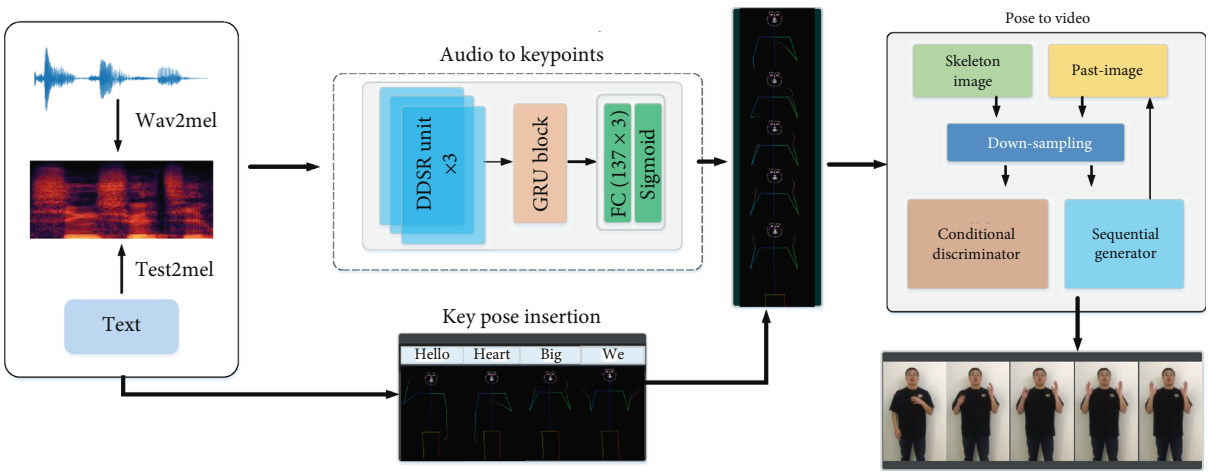


FIGURE 2: Pipeline of our method: the input information can be audio or text. When the audio information is used as the speaker synthesis network input, we convert the audio data into log-mel features and then input the Aud2Kps model to get the pose key points. When the input is text information, it is necessary to use the acoustic model to convert the text information to the log-mel feature as the input of the Aud2Kps network. The following steps are the same as the audio signal input process.

encoding and $[y_0, y_1, \ldots, y_n,]$ is the output open-pose key point vector. The log-mel spectrum feature extracted from audio [25] is a set of 80-dimensional vectors. We designed a DDSR unit to encode the semantic content of features, then input the GRU model to learn the timing features, and finally input the full connection layer and sigmoid activation function to obtain the key point information of the face and human body posture. Our network structure effectively characterizes the audio content information and the correlations between the front and rear time series so that the NME index of the facial key points output by the model is lower. When Aud2Kps maps the audio sequence to the pose sequence, since different parts of the human body have different scales, we need to give them different weights. Therefore, for the body, hands, facial contours, and mouth positions, we set the attention weightsas 1, 10, 50, and 100, respectively. We also use the first-order differential loss between two consecutive poses to ensure that the output pose key points are more smooth and natural.

The MSE loss function L_{MSE} is given by

$$L_{\mathrm{MSE}} = \frac{1}{N}\sum_{i=1}^{N}\left\|y_i - \hat{y}_i\right\|_{l_2}. \tag{1}$$

The first-order temporal differential loss L is given by

$$L_{\mathrm{First-order}} = \frac{1}{N}\sum_{i=1}^{N}\left\|\hat{y}_i - \hat{y}_{i-1}\right\|_{l_1}. \tag{2}$$

At the same time, we use a pose-perception loss function to calculate the content loss between the real and generated pose key points. In most content loss, the VGG network is used as the feature extractor [26, 27], the pose perception loss function uses ST-GCN as the feature extractor of the perception loss function, and the hierarchical representation of the skeleton sequence is formed by using the space-time graph and can be obtained from automatically learn spatial and temporal patterns in the data. We use a dilated residual block in each DDSR unit [28] so that each subsequent layer has a long time span, and the receptive field of the convolutional layer after expansion increases exponentially with the number of layers. This method can effectively increase the sensing receptive field of each output time step and obtain a better long-range correlation. The implementation details of the DDSR unit are shown in Figure 3.

Given a pretrained GCNnetwork φ, we define a collection of layers φ as φ_l. For a training pair (P, M), where P is the ground truth skeleton sequence and M is the corresponding piece of audio, our perceptual loss is

$$L_{\mathrm{Perceptual}} = \sum_{l=1}^{N}\beta_l\left\|\varphi_l(P) - \varphi_l(G(M))\right\|_{l_1}. \tag{3}$$

Here, G is the first-stage Aud2Kps network in our framework. The hyperparameters β_i balance the contribution of each layer l to the loss.

Since the text input will not affect the model efficiency even there is difference in voice characteristics between people, the text input will make the network model more general. Similar to the process of using audio-training Aud2Kps, we convert the text segmentation into phonemes and then use the acoustic model through feature encoding to generate log-mel features as the input of the subsequent speaker synthesis model. We use the open source tacotron2 model to convert the text into a log-mel feature. The following process is the same as the process of audio-to-keypoint.

3.3. Key Pose Insertion. During the model training process, we found that although the Aud2Kps model can synchronize the audio and video content of the speaker very well, the generated character action sequence is too monotonous. This is mainly because the character action sequence is the same at most times in the training set, and the action sequence with posture change is very sparse in the whole

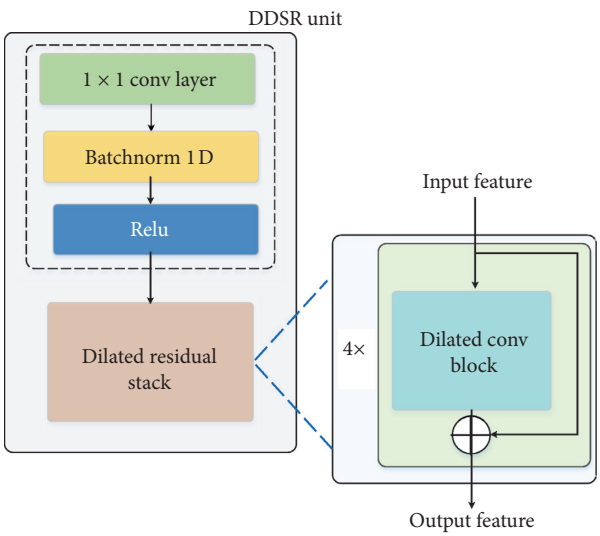


Figure 3: Dilated Depthwise Separable Residual (DDSR) unit network.

training set [29]. In order to make the gesture actions in the synthesized speaker video more expressive and diverse, we designed a gesture sequence dictionary. When the specified keywords appear in the audio content, the corresponding window of the gesture sequence output by Aud2Kps is converted into the specified action, and the posture transformation here uses the posture transformation matrix stored in the posture sequence dictionary.

We select some posture action sequences from the recorded videos and then construct these posture sequences and the corresponding wake-up words into a posture sequence transformation dictionary (composed of transformation matrix). Once the input audio content appears in the dictionary, we will transform the existing pose sequence with a certain probability. The probability between different words may be different. In order to maintain a smooth transition to this pose, we smooth the adjacent frames.

3.4. Pose to Video. We use the vid2vid generator network to convert our generated skeleton images into corresponding speaker videos. After the key points of the human body posture are obtained from the Aud2Kps network, they are visualized as a series of 2D skeleton images, and these 2D images are further fed into the Vid2vid generator network [20] to synthesize the final speaker video. In our network structure, different positions of the human body pay attention to different degrees of importance and people tend to pay more attention to the part of the face and hands. In order to make the vid2vid network pay more attention to the detail texture synthesis of face and hands, we use a separate discriminator network to train the models of face and hand regions to ensure that the discriminator pays more attention to the generated facial and hand details.

4. Experiments

4.1. TalkingPose Dataset. Our audio and video data can be from related speeches or broadcast videos on websites. However, most of the video resources on websites are shot at different times with change of character decorations and clothing styles, which increases uncontrollable factors of samples and increases the difficulty of training. Therefore, we specify speakersto perform audio and video recording. Our speakers read different themes and scripts, and the entire recording time of audio and video is about 2 hours. The video resolution is 1920×1080, and the speed is 30 frames per second.

After recording the video data, the audio data can be directly separated from the corresponding video data. We sample audio data with a sampling rate of 16 kHZ and convert them into log-mel features as the network input. Since audio may have different volume levels, we first normalize its volume through RMS-based normalization [29]. Then, through sparse fast Fourier transform (sfft), the audio is converted from time-domain representation to frequency-domain representation. The value on each frequency represents the energy of the frame of speech signal at the current frequency, and a set of multiple triangular filters are used. The linear spectrum after sfft is processed to obtain 80-dimensional low-dimensional features to simulate the suppression of high-frequency signals by human ears. This method is widely used in speech feature extraction. We use random sampling strategies to expand the dataset for the audio features in the same segment, and the log-mel feature and the posture key point sequence are 1 : 4 as the model input. Figure 2 is a partial example of our dataset.

4.2. Implementation Details. All the models are trained on 8 Nvidia GeForce GTX 1080 Ti GPUs. For the first stage of the Aud2Kps model in our framework, the model is implemented in PyTorch [24] and takes approximately one day to train for 500 epochs. For the hyperparameters, the dimensions of the output channels of the three DDSR units are set to [128, 256, 512], the number of hidden nodes in the GRU timing network is set to 256, and the number of nodes in the final fully connected layer of the network is set to the number of OpenPose parameters 137×3. For the pretraining process of ST-GCN, ST-GCN achieves 49% precision on our TalkingPose dataset. By using the Adam optimizer [30] to minimize the L_2 norm loss of key points in Pytorch, we ensure that the audio features are effectively converted to the corresponding pose key points. The network training batch size is 64, and the learning rate is 0.001. For the second stage that transfers pose to video, the Vid2vid model takes approximately seven days to train for 20 epochs, and the hyperparameters of it adopts the same as [20]. During model training, the data preprocessing part will automatically crop the original video resolution to 1024×1024. Therefore, our results are all 1024×1024 resolution.

4.3. Evaluation Metrics. The task of evaluating speech-driven talking videos is not simple because (1) there is no benchmark dataset to evaluate speech-to-human pose video; (2) the effect of people's speech-driven talking video performance is very subjective, so it is difficult to define model performance. We choose to compare our results with SoTA approaches using the user study. We compare Learning-Gesture [31], neural-voice-puppetry [32], EverybodyDance [33], and Personalized-bodyPose [29] in our user study. In the evaluation metrics of the user study, we refer to the Mean Opinion Score (MOS) [30] of the evaluation index in the text-to-speech (TTS) method [34] to measure the effectiveness of different models. Table 1 shows the MOS of user study for all methods. We get the best overall quality score over the other 4 SOTA methods.

The quantitative model predicts the effect of speaking posture. Even if the people speak the same sentence, he will not perform the same gesture at different moments. It is difficult to judge whether the speech content is correctly converted to the human body posture. However, the facial and mouth shapes of the same sentence are almost the same. Therefore, we evaluate the performance of the model through facial key points. We use the NME indicator [35] to measure the deviation degree that the audio information is converted into corresponding real facial key points. NME is widely used in facial landmark detection to evaluate the quality of models. It is calculated by the average Euclidean distance between predicted and ground truth landmarks, and then it is normalized to eliminate the impact caused by the image size inconsistency. NME for each pose is defined as

$$\mathrm{NME} = \frac{1}{L}\sum_{k=1}^{L}\frac{\left\|p_k - \widehat{p}_k\right\|_2}{d}, \tag{4}$$

where L refers to the number of landmarks, p_k and $\widehat{p}_k$ refer to the predicted and ground truth coordinates of the k_{th} landmark, respectively, and d is the normalization factor, such as the distance of eye centers (interpupil normalization, IPN) or the distance of eye corners (interocular normalization, ION).

To evaluate the effect of pose to video,we use a subjective evaluation method, a user study. In order to evaluate the final output video, we invited 100 participants on the Internet to conduct a subjective test. We showed a total of three videos to participants. Two of them are our synthetic videos, of which, one is a speaker video generated from real human audio, and the other one is a speaker video generated from TTS synthetic audio, and the remaining one is the original real speaker video. These 3 videos are randomly scrambled, and we did not tell the participants the tags behind the videos. Participants need to subjectively rate the quality of these videos, from 1 (strongly disagree) to 5 (strongly agree). The evaluation options include (1) the integrity of the human body; (2) the face of the speaker in the video is clear; (3) the posture of the person in the video looks natural and smooth; (4) the overall visual experience of the video is realistic.

As shown in Table 2, the overall score of our synthetic video four items is 3.795, and the real video is 4.365, which means that the overall effect of our proposed synthetic talking video reaches 86.94% of the real video. It is closer to the real speaker effect in terms of facial details and human body posture integrity. The video score generated by TTS is worse than the voice generation effect, and the reasons are the same as those in Table 3. The main reason is that the synthesized audio has information loss, and hence it is different from the original audio. This loss brings errors into the generated human body postures so that the visual score of the synthesized speaker video is low.

Table 1: Mean Opinion Score (MOS) of 100 participants on 4 questions. *Q*1: completeness of body. *Q*2: the face is clear. *Q*3: the body movement is correlated with audio. *Q*4:overall quality.

	*Q*1	*Q*2	*Q*3	*Q*4
Learning gesture [31]	3.414	3.659	3.914	3.308
Neural-voice-puppetry[32]	3.202	3.840	3.180	3.542
EverybodyDance [33]	3.944	3.662	3.680	3.681
Personalized-bodyPose[29]	3.894	4.011	3.383	3.762
Our method	3.901	4.083	3.526	3.778

Table 2: Mean Opinion Score (MOS) of 100 participants on 4 questions. *Q*1: completenessof body. *Q*2: the face is clear. *Q*3: the body movement is correlated with audio. *Q*4: overall quality.

	*Q*1	*Q*2	*Q*3	*Q*4
Synth.	4.14	4.37	2.92	3.75
TTS	4.10	3.80	2.58	3.39
Real	4.31	4.42	4.33	4.40

4.4. Ablation Study. We use the NME index to evaluate facial key points on the test set. As shown in Table 3, we use different time-length datasets (0.5 h, 1.0 h, 1.5 h, and 2.0 h, respectively) to train the model and observe the impact on the accuracy of pose prediction. In addition, we evaluate the audio data of text synthesis to observe the impact of sound changes on the results, use text to train and test the network, and compare the results with the audio results. Finally, we compare the training using only the GRU network with that using our network structure.

From Table 3, we can notice the following. (1) After the audio training set is increased to 1.5 h, the model benefit will not be great by increasing the dataset, but the model effect can also be improved by further increasing the amount of data on the text training set. (2) From the model indicators obtained from audio and text data, it can be seen that the effect of audio is worse than that of text , indicating that the audio conversion to the key points of the face is more accurate. (3) The audio data synthesized by text is tested on the model. The effect is not as good as the original audio mainly because the synthesized audio has information loss, and hence it is different from the original audio. (4) Using the DDSR unit network model is better than only using the GRU network structure as feature extractor. Although only using the GRU network can capture the correlation between the front and rear frames, the feature representation ability is

Table 3: Evaluation metrics used NME (%) on facial landmarks (lower is better).

	Orig.	Only-GRU	TTS-mel	Text
0.5	4.925	5.673	5.871	5.693
1.0	4.921	5.640	5.885	5.690
1.5	4.853	5.644	5.877	5.614
2.0	4.907	5.647	5.829	5.607

weak. The combination of the DDSR unit and the GRU can make up for this shortcoming.

To prove the effectiveness of our key pose insertion method, we conducted another user study. In this study, we simply presented a pair of composite videos with and without inserting key poses. Participants only need to evaluate which of the two videos is more natural and realistic. From the final user rating, it is shown that the synthesized video with gesture actionsbeing inserted into its existing posture sequence scored 81.3% and the synthesis video without the key frame poses only received 18.7% of votes. This illustrates the effectiveness of inserting pose key points to enrich speech-driven talking video synthesis.

5. Conclusion and Future Work

In this work, we propose a new method to generate realistic talking video from audio information. We sample the audio data randomly and use a more effective network structure to learn the audio content and timing correlation.We use first-order differential loss and pose perception loss to optimize the network output so that the face and pose key points obtained by audio conversion are smoother and the index performance is better. At the same time, by inserting a specified action frame into the synthesized human pose sequence window, the synthesized speaker's action posture is more natural and realistic. Our objective and subjective evaluation comparison results are very competitive over the existing methods. Our current method has good results in single-speaker scenarios. In multispeaker audio-video conversion tasks, we use TTS technology to convert speech to text to eliminate the inconvenience caused by voice ID information. In the future, we will further explore the work related to multispeaker to multitarget character video synthesis.

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of PR China (42075130).

References

[1] T. Karras, T. Aila, S. Laine, A. Herva, and J. Lehtinen, "Audio-driven facial animation by joint end-to-end learning of pose and emotion," *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 1–12, 2017.

[2] Y. Zhou, Z. Xu, C. Landreth, E. Kalogerakis, S. Maji, and K. Singh, "VisemeNet: audio-driven animator-centric speech animation," *ACM Transactions on Graphics*, vol. 37, no. 4, pp. 1–10, 2018.

[3] S. E. Eskimez, R. K. Maddox, C. Xu, and Z. Duan, "Generating talking face landmarks from speech," in *Proceedings of the International Conference on Latent Variable Analysis and Signal Separation*, pp. 372–381, Guildford, UK, June 2018.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, NV, USA, June 2016.

[5] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, Salt Lake City, UT, USA, June 2018.

[6] K. ChoB. Van Merriënboer et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, Doha, Qatar, October 2014.

[7] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proceedings of the Association for the Advancement of Artificial Intelligence*, pp. 1–9, New Orleans, LA, USA, February 2018.

[8] J. B. Estrach, P. Sprechmann, and Y. LeCun, "Super-resolution with deep convolutional sufficient statistics," in *Proceedings of the 4th International Conference on Learning Representations, ICLR*, San Juan, Puerto Rico, May 2016.

[9] K. Choi, Y. Luo, and J.-N. Hwang, "Hidden markov model inversion for audio-to-visual conversion in an mpeg-4 facial animation system," *The Journal of VLSI Signal Processing*, vol. 29, no. 1/2, pp. 51–61, 2001.

[10] D. Cosker, D. Marshall, P.L. Rosin, and Y. Hicks, "Speech driven facial animation using a hidden markov coarticulation model," in *Proceedings of the 17th International Conference on Pattern Recognition (ICPR)*, pp. 128–131, Cambridge, UK, August 2004.

[11] D. Cosker, D. Marshall, P. Rosin, and Y. Hicks, "Video realistic talking heads using hierarchical non-linear speech-appearance models," in *Proceedings of the Mirage 2003*, Le Chesnay-Rocquencourt, France, March 2003.

[12] L. D. Terissi and J. C. Gómez, "Audio-to-visual conversion via HMM inversion for speech-driven facial animation," in *Proceedings of the Brazilian Symposium on Artificial Intelligence*, pp. 33–42, Salvador, Brazil, October 2008.

[13] L. Xie and Z.-Q. Liu, "A coupled HMM approach to video-realistic speech animation," *Pattern Recognition*, vol. 40, no. 8, pp. 2325–2340, 2007.

[14] X. Zhang, L. Wang, G. Li, F. Seide, and F.K. Soong, "A new language independent, photo-realistic talking head driven by voice only," in *Interspeech*, pp. 2743–2747, Springer, Berlin, Germany, 2013.

[15] M. Brand, "Voice puppetry," in *Proceedings of the 26th Annual Conference on Com- Puter Graphics and Interactive Techniques*, pp. 21–28, Los Angeles, CA, USA, August 1999.

[16] S. Suwajanakorn, S.M. Seitz, and I. Kemelmacher-Shlizerman, "Synthesizing obama: learning lip sync from audio," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, p. 95, 2017.

[17] H.X. Pham, S. Cheung, and V. Pavlovic, "Speech-driven 3D facial animation with implicit emotional awareness: a deep learning approach," in *Proceedings of the 1st DALCOM Workshop, CVPR*, Guildford, UK, 2017.

[18] H. X. Pham, Y. Wang, and V. Pavlovic, "End-to-end learning for 3D facial animation from speech," in *Proceedings of the 20th ACM International Conference on Multimodal Interaction*, pp. 361–365, Boulder, CO, USA, October 2018.

[19] Y. Zhou, Z. Li, and S. Xiao, "Auto-conditioned recurrent networks for extended complex human motion synthesis," in *Proceedings of theInternational Conference on Learning Representations*, Vancouver, Canada, May 2018.

[20] T. C. Wang, M. Y. Liu, and J. Y. Zhu, "Video-to-video synthesis," in *Proceedings of theConference on Neural Information Processing Systems (NeurIPS)*, Montreal, Canada, December 2018.

[21] A. ShysheyaE. Zakharov et al., "Textured neural avatars," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2387–2397, San Juan, PR, USA, June 2019.

[22] J. Shen, R. Pang, R. J. Weiss et al., "Natural TTS synthesis by conditioning wavenet on MEL spectrogram predictions," in *Proceedings of the2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4779–4783, Calgary, Canada, April 2018.

[23] Z. Cao, G. Hidalgo Martinez, T. Simon, S.-E. Wei, and Y. A. Sheikh, "OpenPose: realtime multi-person 2D pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, p. 1, 2019.

[24] S. E. Wei, V. Ramakrishna, and T. Kanade, "Convolutional pose machines," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4724–4732, Las Vegas, NV, USA, June 2016.

[25] K. Kumar, R. Kumar, and T. De Boissiere, "Melgan: generative adversarial networks for conditional waveform synthesis," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 14910–14921, Vancouver, Canada, 2019.

[26] Q. Chen and V. Koltun, "Photographic image synthesis with cascaded refinement networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1511–1520, Venice, Italy, October 2017.

[27] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Proceedings of the European Conference on Computer Vision*, pp. 694–711, Amsterdam, Netherlands, October 2016.

[28] S. Mehta, M. Rastegari, and A. Caspi, "Espnet: efficient spatial pyramid of dilated convolutions for semantic segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 552–568, Munich, Germany, September 2018.

[29] M. Liao, S. Zhang, and P. Wang, "Speech2video synthesis with 3D skeleton regularization and expressive body poses," in *Proceedings of the Asian Conference on Computer Vision*, Kyoto, Japan, December 2020.

[30] R. Skerry-Ryan and E. Battenberg, "Towards end-to-end prosody transfer for expressive speech synthesis with tacotron," in *Proceedings of the 35th International Conference on Machine Learning*, pp. 4693–4702, Stockholm Sweden, July 2018.

[31] S. Ginosar, A. Bar, G. Kohavi, C. Chan, A. Owens, and J. Malik, "Learning individual styles of conversational gesture," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3497–3506, Long Beach, CA, USA, June 2019.

[32] J. Thies, M. Elgharib, and A. Tewari, "Neural voice puppetry: audio-driven facial reenactment," in *Proceedings of the European Conference on Computer Vision*, pp. 716–731, Glasgow, UK, August 2020.

[33] C. Chan, S. Ginosar, T. Zhou, and A.A. Efros, "Everybody dance now," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5933–5942, Seoul, Republic of Korea, October 2019.

[34] J. M. Valin and J. Skoglund, "LPCNet: improving neural speech synthesis through linear prediction," in *Proceedings of theICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5891–5895, Brighton, UK, May 2019.

[35] A. Bulat and G. Tzimiropoulos, "How far are we from solving the 2D& 3D face alignment problem?(and a dataset of 230,000 3D facial landmarks)," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1021–1030, Venice, Italy, October 2017.

Hindawi
Complexity
Volume 2020, Article ID 5831632, 10 pages
https://doi.org/10.1155/2020/5831632

Review Article

A Multi-Index Generative Adversarial Network for Tool Wear Detection with Imbalanced Data

Guokai Zhang,[1] **Haoping Xiao**,[2] **Jingwen Jiang**,[3] **Qinyuan Liu**,[2] **Yimo Liu**,[4] **and Liying Wang**[5]

[1]*School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai, China*
[2]*Department of Computer Science and Technology, Tongji University, Shanghai, China*
[3]*Technology Transformation Center, Shanghai Electric Group Co., Ltd. Central Academe, Shanghai, China*
[4]*School of Software, Tongji University, Shanghai, China*
[5]*Shanghai Leosue Network Technology Co., Ltd., Shanghai, China*

Correspondence should be addressed to Qinyuan Liu; liuqy@tongji.edu.cn

Received 21 June 2020; Revised 30 October 2020; Accepted 20 November 2020; Published 7 December 2020

Academic Editor: Min Xia

The scarcity of abnormal data leads to imbalanced data in the field of monitoring tool wear conditions. In this paper, a novel multi-index generative adversarial network (MI-GAN) is proposed to detect the tool wear conditions subject to imbalanced signal data. First, the generator in the MI-GAN is trained to produce fake normal signals, and the discriminator computes scores of testing signals and generated signals. Next, the generator detects abnormal signals based on the performance of imitating testing signals, and the discriminator will compute the scores of testing signals and generated signals. Subsequently, two indexes, i.e., L_2-norm and temporal correlation coefficient (CORT), are put forward to measure the similarity between generated signals and testing signals. Finally, our decision-making function further combines L_2-norm and CORT with two discriminator scores to determine the tool conditions. Experimental results show that our method obtains 97% accuracy in tool wear detection based on imbalanced data without manual feature extraction, which outperforms traditional machine learning methods.

1. Introduction

The past decades have witnessed the rapid development of equipment-manufacturing technologies. With the widespread applications of computerized numerical control (CNC) machines, the productivity and accuracy in the manufacturing processes have been significantly enhanced. As an important part of CNC machines, the conditions of the cutting tools are critical to the quality of products. However, since the cutting tools are usually subject to extremely severe rubbing environment, tool wear is inevitable in the manufacturing processes. Due to the frequent friction and extrusion, the cutting tools gradually wear out, lose sharpness, and become dull. The wear of cutting tools might increase the cutting forces and temperature, degrade the machine surface texture, and even lead to fatal failures to normal operation of the manufacturing systems in some extreme cases. Therefore, designing effective real-time tool wear monitoring techniques that can guarantee high quality in the manufacturing process has received much attention from both academia and industry [1–3].

Many attempts have been made to detect tool wear in recent years, among which image-based methods are the most widely used approaches. For example, the authors in [4–6] have developed an image-based online tool condition detection technique by using optical methods. However, it should be noted that such an image-based method is usually vulnerable to the manufacturing environment as the cutting fluid, chips, and dirt might hinder the image acquisition of the workpieces, and hence, the tool conditions can only be measured at the end of one batch. To handle such an issue, some online tool condition detection methods are proposed based on signal analysis. As is well recognized, there exists a certain relationship between the tool wear condition and the

manufacturing signals, e.g., vibration, acoustic, and current signals. These signals would present different features when the cutting tool works in different conditions. By deploying specific sensors, manufacturing signals can be easily collected and then exploited to determine the tool wear condition in a real-time manner.

Up to now, various methods have been proposed on tool wear monitoring techniques based on signal analysis. To be specific, the authors in [7] have utilized feedback current signals to classify the tool condition. The performance and accuracy of several pattern recognition techniques, including support vector machine (SVM), linear discriminant analysis (LDA), K-nearest neighbor (KNN), neural network (NN) with one hidden layer, naïve Bayes (NB), and decision trees (DT), have been compared. It has been concluded that, among these classification techniques, LDA and SVM are the most effective ones. Moreover, a comparative study has been conducted in [8] which demonstrates that the predictive capability of random forests (RFs) outperforms artificial neural networks (ANNs) with one hidden layer and support vector regression (SVR). However, all these traditional techniques [7–13] highly rely on the features manually extracted from raw data, which need to analyze characteristics of the manufacturing signals and extract reliable features based on prior knowledge of specific tasks, and thus are cumbersome. Consequently, it is of practical significance to develop an easy-to-use method without the feature extraction procedure.

It should be pointed out that deep learning is a promising technique capable of automatically searching information features behind the raw data. In [14–17], the feasibility of convolutional neural networks (CNNs) on extracting information features has been presented. In fact, one can convert input signals into images using Gramian Angular Summation Fields and then leverage CNN to classify the tool wear condition [18]. In [19], a stacked sparse autoencoder neural network has been presented to diagnose the tool wear conditions. By decomposing original 1D signals to reconstruct 2D ones, CNNs can accurately predict the tool wear condition [20]. However, there are some critical shortcomings of the CNN-based tool condition detection [14–17]. For example, a large labeled dataset is required to improve the performance of the CNN, and the signal-to-image conversion procedure might result in a great information loss of the raw signals [18]. In addition, for the majorities of the industrial processes, the normal and abnormal labels of the data are imbalanced. The low failure rate of the manufacturing machines leads to the scarcity of abnormal data, and with limited abnormal data, it is almost impossible to describe the accurate characteristics of all fault types.

Very recently, generative adversarial networks have been utilized to solve imbalanced data problems in unsupervised anomaly detection. To be specific, the authors in [21] have put forward a generative adversarial network with an encoder-decoder-encoder three-subnetwork generator to handle imbalanced industrial time series, but an obvious shortcoming lies in that it still has to manually extract features. Moreover, some other GAN-based methods such as AnoGAN and GAN-AD have also been proposed to achieve detection targets [22, 23]. In these previous works, an anomaly score is established to detect the abnormal signals by directly combining the residual loss and the discriminator loss. However, it is noted that the magnitude of the residual loss is usually different from that of the discriminator loss, and thus, a direct combination of these two losses would not lead to a satisfactory detection.

Motivated by the above discussion, in this paper, we propose a novel multi-index GAN (MI-GAN) method for tool wear detection involving imbalanced data. To deal with the challenges stemming from data imbalance, the generator of the proposed MI-GAN focuses on learning the characteristics of the normal signals, and the discriminator outputs the scores which represent the fake level of the input signals. Such a network architecture can effectively detect the tool condition by learning how to generate and distinguish normal signals. Especially, a novel multi-index decision-making function which incorporates L_2-norm, temporal correlation coefficient (CORT) [24], the score of the input signal, and the score of the generated signal is developed to aggregate more representative criteria. In the end, we use the temporal convolution networks (TCNs) as the backbone of our generator and discriminator networks to encode more temporal and sequential features, which could further leverage the performance of the model instead of the empirical feature definition. We sample real current signals from a CNC machine in different manufacturing processes and then conduct the tool wear detection experiments. The experimental results demonstrate that our designed MI-GAN could achieve competitive performance on the accuracy, precision, recall, and F-measure in detecting the abnormal tool conditions. Overall, the main contributions of this paper can be highlighted as follows: (1) a novel MI-GAN which can generate and distinguish normal signals is established to solve imbalanced data in monitoring tool wear conditions; (2) a decision-making function with multi-index is developed to aggregate more representative criteria; and (3) the feature extraction ability is enhanced by applying the TCN as the backbone of our MI-GAN generator and discriminator networks without predefined feature selection.

The rest of the paper is organized as follows. Section 2 introduces the methodology of the proposed MI-GAN including the data acquisition procedure, the architecture of the MI-GAN, the training process, and the multi-index decision-making algorithm. In Section 3, we carry out the experimental results to evaluate the performance of our method. Section 4 finally draws conclusions.

2. Structure of Monitoring Systems

In the manufacturing processes of CNC machines, it is more suitable to monitor the state of the tool by sampling the current signals of the spindle motor and analyzing their real-time characteristics. As shown in Figure 1, the spindle transmission system consists of a motor, spindle, timing belt, drive, etc. The main shaft is connected to the motor through a belt, the machine tool sends a control signal to the drive which outputs the spindle motor current to drive the motor

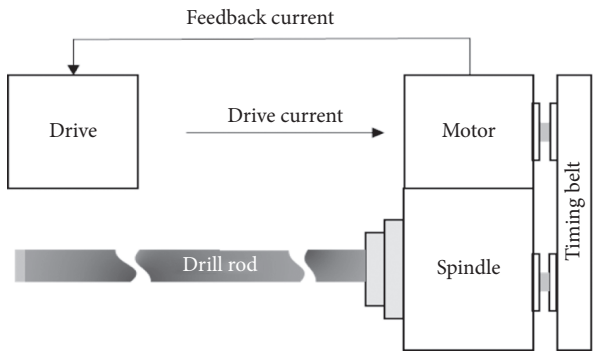


Figure 1: The overall structure of the spindle transmission system of the CNC machine tool.

to rotate, and the spindle is derived by the motor via a timing belt. Since the spindle motor current is positively related to the cutting force, the change of the real-time spindle current could reflect the change of the cutting force. With the gradual wear of the tool during the machining process, the cutting performance of the tool will gradually decrease and thus requires greater cutting force which leads to the increase of motor current. Therefore, the current signal of the spindle is utilized to detect the tool conditions throughout the paper.

3. Methodology

A detailed introduction of the proposed MI-GAN will be presented in this section. Generally speaking, the procedure of tool condition detection based on MI-GAN can be roughly divided into the following two stages.

In stage 1, the normal signals are exploited to train the generative adversarial network. The generator tries to synthesize normal signals by minimizing generator loss. Generated signals and normal signals are fed into the discriminator in sequence to train the network. When the training losses converge, the generator can stably generate plausible normal signals, and the discriminator can measure the fake-level scores of actual signals and generated signals. Here, actual signals denote the signals in the dataset.

In stage 2, we will make full use of our pretrained model to calculate four specific indexes (i.e., L_2-norm, CORT, and two fake-level scores, which will be further clarified in Section 3.4). First, by inverting input signals to specific noises z, the pretrained generator is able to generate signals similar to input signals. Next, the values of indexes L_2-norm and CORT can be calculated between the generated signals and the input signals. The fake-level scores of the generated signals and the input signals will be output from the pretrained discriminator. Furthermore, the mean and standard deviation of these four indexes are calculated to determine the detecting thresholds. Eventually, the model compares four indexes of the testing signal with the thresholds to determine either normal signal or abnormal. A brief overview is provided to help understand our method in Figure 2, and all the technical details will be explained in the following sections.

3.1. Data Acquisition. The dataset used is the spindle current signals collected by the CNC machine tool when drilling 3 holes in a workpiece. The sampling frequency of the current signals is 10 Hz, and each 130 sampling point data collected is recorded as a set of signals. In each manufacturing process, the CNC machine tools are monitored in time, and their tool condition (i.e., normal or abnormal) was recorded as a label attached with each set. The dataset can be denoted as $C = \{C_1, C_2, \ldots, C_N\}$, where $C_i = \{X_{i1}, X_{i2}, \ldots, X_{in}\}$ is time series data with the total length n, and X_{ik} is the current value of C_i at sampling instant k. We divide our dataset into two subsets, i.e., training dataset and testing dataset. The training dataset includes 800 normal signals, and the testing dataset contains 287 normal signals and 11 abnormal signals.

3.2. Architecture of the MI-GAN. The schematic diagram of the MI-GAN architecture is illustrated in Figure 3, which consists of a generator and a discriminator. The generator projects the random noise space to the normal signal space so as to confuse the discriminator. It takes a vector of 130 random numbers independently drawn from a uniform distribution as the input and outputs a fake normal signal of length 130. Both generated signals and normal signals are fed into the discriminator to obtain fake-level scores, which indicate how fake the input signals are. A higher fake-level score indicates that the input signal is more likely to be an abnormal one, and vice versa. Notice that the TCN residual block (as depicted in Figure 4) has excellent sequence modeling performance [25]. Consequently, the generator first utilizes six TCN residual blocks to extract the features of input noise vector z and then a linear layer to synthesize the plausible signal. Similarly, the TCN residual blocks are also exploited as a part of our discriminator, and two linear layers are added to compute fake-level scores.

3.3. Training Generative Adversarial Networks. The classical training method for the generative adversarial networks is to optimize Jensen–Shannon (JS) or Kullback–Leibler divergence between the model distribution and real-data distribution. However, it is pointed out in [26] that such methods are unreliable and might lead to training instability and mode collapse. To avoid this problem, we use Wasserstein divergence [27] as our min-max objective as follows:

$$\min_G \max_D \mathbb{B}_{G(z)\sim\mathbb{P}_g}[D(G(z))] - \mathbb{B}_{C\sim\mathbb{P}_r}[D(C)] - 2\mathbb{B}_{\tilde{C}\sim\mathbb{P}_u}\left[\left\|\nabla_{\tilde{C}} D(\tilde{C})\right\|^6\right], \tag{1}$$

where $\mathbb{P}_g$ is the distribution of generated signals, $\mathbb{P}_r$ is the distribution of real signals, $\mathbb{P}_u$ is a Radon probability measure, z represents random noises, C indicates the normal signals, and $\tilde{C}$ is a sequence sampled from $\mathbb{P}_u$. $G(.)$ and $D(.)$ represent the generator and discriminator, respectively.

Hence, the generative adversarial networks are trained by optimizing the generator loss L_G and the discriminator loss L_D given in (2) and (3) as follows:

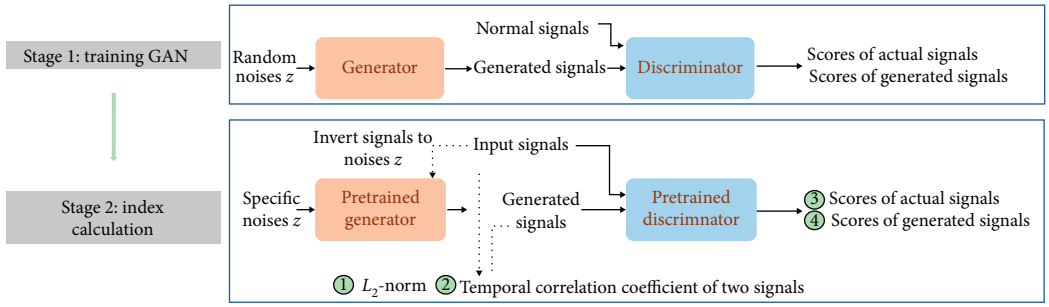


Figure 2: The overall framework of the MI-GAN. In stage 1, the generative adversarial network is trained. Stage 2 makes full use of the pretrained model to calculate four specific indexes.

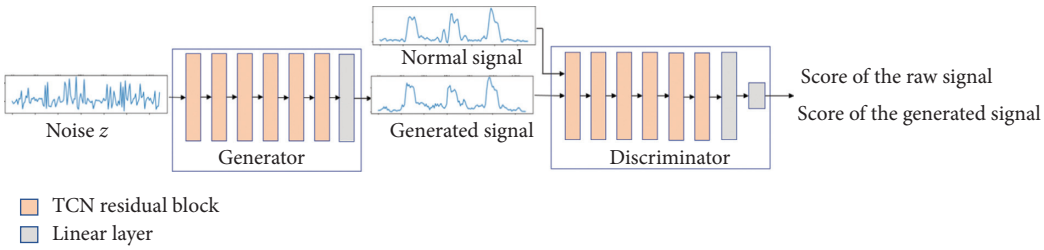


Figure 3: The schematic diagram of the MI-GAN architecture. The generator projects the random noise space to the normal signal space, and the discriminator calculates the final fake-level scores of the abnormal signals.

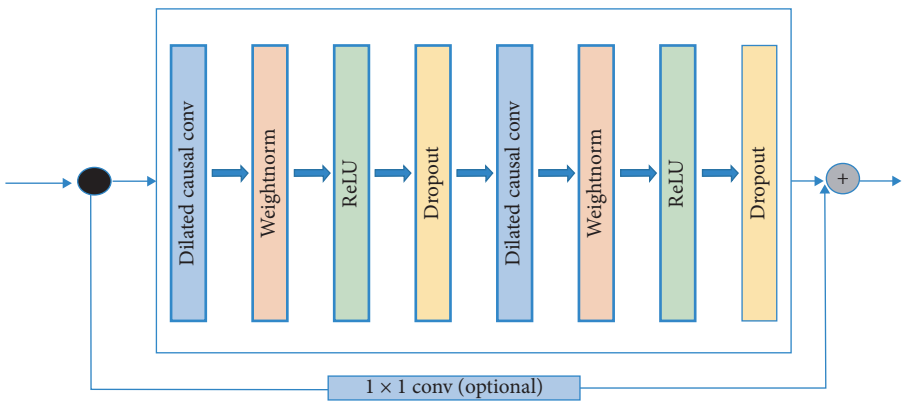


Figure 4: The architecture of TCN residual blocks.

$$L_G = \mathbb{B}_{G(z)\sim\mathbb{P}_g}[D(G(z))], \tag{2}$$

$$L_D = \mathbb{B}_{C\sim\mathbb{P}_g}[D(C)] - \mathbb{B}_{G(z)\sim\mathbb{P}_r}[D(G(z))] + 2\mathbb{B}_{\widetilde{C}\sim\mathbb{P}_u}\left[\left\|\nabla_{\widetilde{C}} D(\widetilde{C})\right\|^6\right], \tag{3}$$

where $\widetilde{C} = (1-\mu)C + \mu G(z)$ and μ is a nonnegative scalar belonging to $[0, 1]$.

Inspired by [21, 23], only the normal signals are input into the discriminator for the sake of encouraging the MI-GAN to generate fake normal signals. The generator produces fake normal signals to confuse the discriminator by minimizing L_G, and the discriminator is trained to compute the fake-level score to distinguish normal signals and fake normal signals by minimizing L_D. Furthermore, the Adam algorithm [28] is adopted to optimize the generator and discriminator losses. In the implementation, we choose the same weights for C and $G(z)$ to compute $\widetilde{C}$, i.e., $\mu = 0.5$.

3.4. Threshold Determination. We utilized both generator and discriminator to monitor tool wear conditions. To make full use of the generator, the input signals will be projected to the latent space using our pretrained generator. Referring to the inversion technique in [29], we apply Algorithm 1 in Table 1 to search specific noises z as the input of the generator, whose corresponding generated signals $\widetilde{C}$ are similar to the input signals C. This inversion technique uses the RMSprop optimization algorithm [30] to minimize the mean square error of generated signals and input signals.

Since the generator captures only the normal pattern, it is difficult to output fake abnormal signals. When the input signals are abnormal, the generated signals will be dissimilar

Table 1: Specific noise-searching algorithm.

Algorithm 1
Require: batch size m, input signals $C = (C_1, C_2, \ldots, C_m)$, pretrained generator G, training iterations n, and learning rate α.
(1) Sample noises $z = (z_1, z_2, \ldots, z_m)$ from standard normal distribution.
(2) **for** $k = 1$ to t do:
(3) Generate a batch of fake normal signals: $\widetilde{C} = G(z)$.
(4) Calculate the mean square error between C and $\widetilde{C}$ as $L \leftarrow 1/m \sum_{i=1}^{m} \lVert C_i - \widehat{C}_i \rVert^2$, where $\widetilde{C}_i = G(z_i)$ is the ith element in $\widehat{C}$.
(5) $z \leftarrow \text{RMS prop}(L, \alpha)$.
(6) **end for**
(7) **return** the specific noises z.

with input signals. Therefore, the generated signals will fall into two categories: one is similar signals whose inputs are normal signals, and the other one is dissimilar signals whose inputs are abnormal signals.

Denote $\widehat{C}_i = (\widehat{X}_{i1}, \widehat{X}_{i2}, \ldots, \widehat{X}_{in})$, $X_k = (X_{1k}, X_{2k}, \ldots, X_{mk})$, and $\widehat{X}_k = (\widehat{X}_{1k}, \widehat{X}_{2k}, \ldots, \widehat{X}_{mk})$. In order to measure the similarity between generated signals and input signals, we introduce L_2-norm (i.e., $L(C, \widehat{C})$) and first-order CORT (i.e., $\text{CORT}(C, \widehat{C})$) as follows:

$$L(C, \widehat{C}) = \sqrt{\sum_{k=1}^{n} \left\lVert X_k - \widehat{X}_k \right\rVert^2}, \tag{4}$$

$$\text{CORT}(C, \widehat{C}) = \frac{\sum_{k=1}^{n-1} (X_{k+1} - X_k)\left(\widehat{X}_{k+1} - \widehat{X}_k\right)^T}{\sqrt{\sum_{k=1}^{n-1} \left\lVert X_{k+1} - X_k \right\rVert^2} \sqrt{\sum_{k=1}^{n-1} \left\lVert \widehat{X}_{k+1} - \widehat{X}_k \right\rVert^2}}. \tag{5}$$

In fact, L_2-norm in (4) indicates the distance between generated signals and input signals, and the first-order CORT in (5) evaluates their growth behavior similarity. A larger L_2-norm or a smaller CORT means a larger dissimilarity between generated signals and input signals. That is, the input signals are more probable to be abnormal.

Furthermore, we also utilize our pretrained discriminator to help distinguish whether the input signal is normal. The output of our discriminator can be regarded as a fake-level score. Since we only feed normal data in the training stage, a higher fake-level score represents the signal is less likely to be a normal one. Therefore, the discriminator will output a high fake-level score for abnormal signals. The fake-level scores of normal signals will be lower than those of abnormal signals, and their corresponding generated signals will have the same situation. By now, we have put forward four indexes to help identify the tool wear condition, which are L_2-norm, CORT, the fake-level scores of input signals, and the fake-level scores of generated signals.

Given these indexes, four thresholds could be established to distinguish the normal and abnormal signals. In order to determine the threshold values, we need to calculate four indexes of the normal signals in the training dataset. Subsequently, the mean and standard deviation of the values of the indexes are computed, which are critical for identifying abnormal inputs. Since CORT is inversely related to the possibility of signals to be abnormal and the other three indexes are positively correlated, the thresholds of L_2-norm of normal signals and two fake-level scores are defined as their mean plus standard deviation, whereas the threshold of CORT is denoted as its mean value minus standard deviation. These thresholds are based on statistic principles and thus can classify the input signals effectively.

3.5. Multi-Index Decision-Making Function. We develop a decision-making function to reach the detection target. For each input signal, the function uses the pretrained model and equations (4) and (5) to obtain four indexes. At first, these indexes are uniformed by subtracting their corresponding thresholds to determine whether the signal is abnormal or not. For uniformed L_2-norm and two fake-level scores, positive values indicate that the inputs are more likely to be abnormal, and their magnitude describes the corresponding possibility. For uniformed CORT, negative values indicate a higher probability of input signals to be abnormal. In other words, the less similar the generated signals are, the smaller the CORT is. After the uniformed procedure, these indexes are further scaled into the same range by dividing their standard deviation, respectively. Finally, we integrate all the scaled indexes to a comprehensive score, based on which one can infer the tool wear condition. To sum up, our multi-index decision-making algorithm is elaborated in Table 2.

4. Experimental Results

4.1. Evaluation Metrics. As the current signal data are imbalanced, we use accuracy, weighted average recall (W-recall), precision, and F-measure to evaluate the overall performance of our detection method. Denote TP, FP, TN, and FN as true positive, false positive, true negative, and false negative, respectively. For each class, the recall, precision, and F-measure can be calculated as follows:

$$\begin{aligned} \text{Recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}}, \\ \text{Precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}}, \\ F - \text{measure} &= \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \end{aligned} \tag{6}$$

Moreover, the W-recall, W-precision, and W-F metrics are defined as follows:

Table 2: Multi-index decision-making algorithm.

Algorithm 2
Require: mean of L_2-norm, CORT, fake-level score of normal signals, and fake-level score of the generated signal, denoted as L_m, CORT_m, $F_m(n)$, and $F_m(g)$. Standard deviation of L_2-norm, CORT, fake-level score of normal signals, and fake-level score of the generated signal, denoted as L_{std}, $\mathrm{CORT}_{\mathrm{std}}$, $F_{\mathrm{std}}(n)$, and $F_{\mathrm{std}}(g)$. Input signals C, pretrained Generator G, and pretrained Generator D.
(1) Invert C to the latent space to search noises z using Algorithm 1.
(2) Generate a batch of signals: $\widehat{C} = G(z)$.
(3) Calculate the L_2-norm and CORT between C and $\widehat{C}$ based on (4) and (5).
(4) Compute the fake-level scores of C and $\widehat{C}$, i.e., $D(C)$ and $D(\widehat{C})$, based on the discriminator
(5) Subtract thresholds:
(6) $Index_1 = L(C, \widehat{C}) - (L_m + L_{std})$
(7) $Index_2 = \mathrm{CORT}(C, \widehat{C}) - (\mathrm{CORT}_m - \mathrm{CORT}_{std})$
(8) $Index_3 = D(C) - (F_m(n) + F_{std}(n))$
(9) $Index_4 = D(\widehat{C}) - (F_m(g) + F_{std}(g))$
(10) Scale the indexes:
(11) $Index_1 = Index_1 / L_{std}$
(12) $Index_2 = -Index_2 / CORT_{std}$
(13) $Index_3 = Index_3 / F_{std}(n)$
(14) $Index_4 = Index_4 / F_{std}(g)$
(15) Combine the indexes:
(16) $Score = Index_1 + Index_2 + Index_3 + Index_4$
(17) **If** $score <= 0$:
(18) Input signals are normal.
(19) **Else:**
(20) Input signals are abnormal.

$$W - \mathrm{recall} = \frac{\sum_{i=1}^{s} (c_i \times \mathrm{Recall}_i)}{\sum_{i=1}^{n} c_i},$$
$$W - \mathrm{precision} = \frac{\sum_{i=1}^{s} (c_i \times \mathrm{Precision}_i)}{\sum_{i=1}^{s} c_i}, \quad (7)$$
$$W - F = \frac{\sum_{i=1}^{s} (c_i \times F\text{-measure}_i)}{\sum_{i=1}^{s} c_i},$$

where c_i is the number of instances in class i and s is the number of total classes. Recall_i, $\mathrm{Precision}_i$, and F-$\mathrm{measure}_i$ are the recall, precision, and F-measure of the ith class, respectively.

4.2. Generation Capability of the Generator. To evaluate the performance of the generator, we compare the generation ability of the generator when the inputs are normal signals and abnormal signals. As we utilize only normal signals to train the MI-GAN, it is effective to obtain inverted noises z and produce similar signals to the normal signals. Nevertheless, the pretrained generator never learns the features of abnormal signals and thus cannot produce fake abnormal signals even when the inputs are abnormal signals. In other words, when the input signals are abnormal, the generated signals will be dissimilar with input signals. Therefore, the generated signals will fall into two categories: one is similar signals whose inputs are normal signals, and the other one is dissimilar signals whose inputs are abnormal signals.

Figure 5 presents the generation ability of our pretrained generator. In Figure 5, we input two types of signals (i.e., normal and abnormal ones) and utilize Algorithm 1 to search specific noises z. Since the reconstruction capability for normal and abnormal signals is different, the generated signals will fall into two categories, i.e., similar signals and dissimilar signals. Using this characteristic of the generator, one can classify different types of input signals to achieve the tool wear conditions.

4.3. Discrimination Ability of the Discriminator. In this paper, instead of computing real or fake probability, the discriminator outputs a fake-level score, which can interpret how fake the signals are. To demonstrate the discrimination ability of the discriminator, we put the mean and standard deviation of the fake-level scores in Table 3.

According to Table 3, we can see that the mean fake-level scores of actual signals (−129.47 and −88.08) are lower than those of generated signals (−78.30 and −49.11), which indicates actual signals are more real than generated signals. In addition, normal signals have a lower mean fake-level score (−129.47) than that of abnormal signals (−88.08) since the generator can capture only the features of the normal signals. As for generated signals, it is straightforward to expect that the signals dissimilar to abnormal input signals have a higher mean fake-level score (−49.11) than that of the signals similar to normal input signals (−78.30). The standard deviation of abnormal signals is obviously larger than the remaining ones since the abnormal signals have more diverse modes than other ones.

4.4. Distribution of Four Indexes. To prove the effectiveness of four indexes, their respective distributions are plotted in Figure 6. We consider three different data types including normal signals in the training dataset, normal signals in the

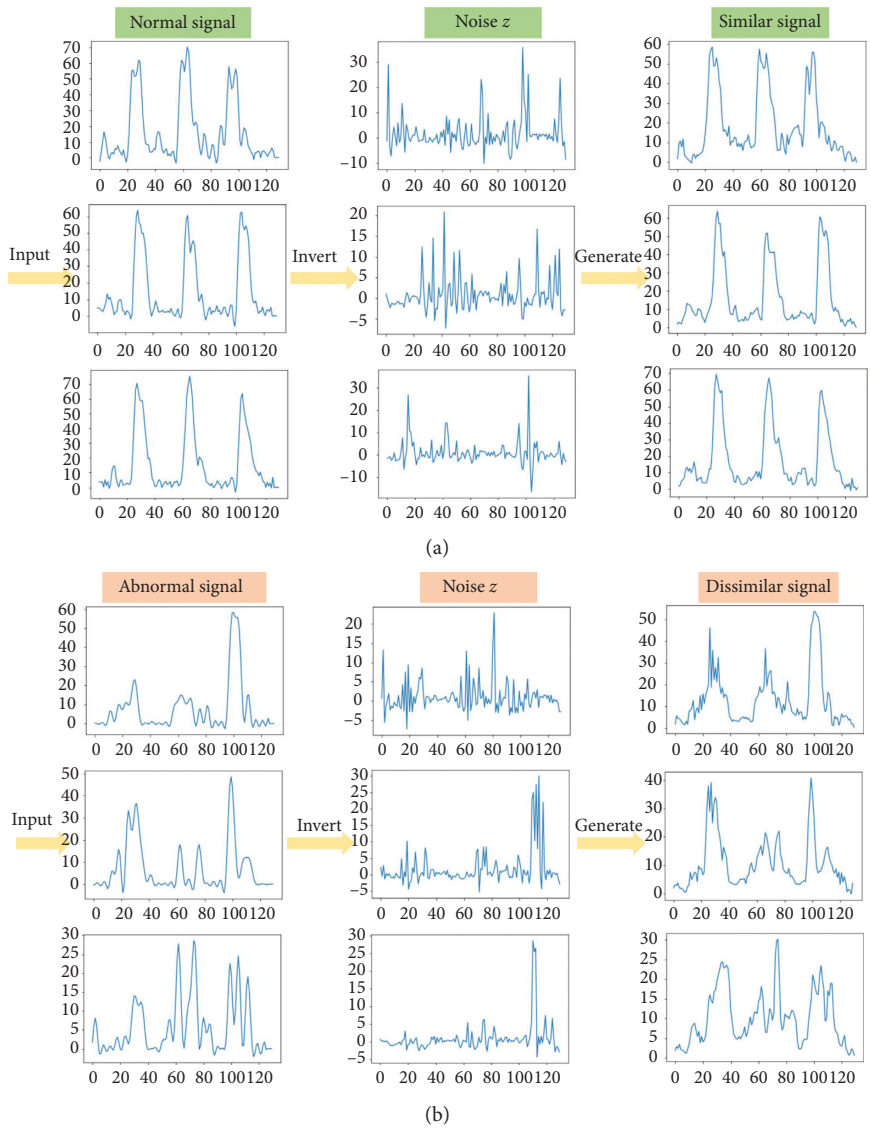


Figure 5: Invert input signals to generate new signals. (a) If the input signal is normal, the generated signal is similar to the input signal. (b) If the input signal is abnormal, the generated signal is dissimilar to the input signal.

Table 3: The mean and standard deviation of the scores.

Types	Mean	Standard deviation
Scores of normal signals	−129.47	39.61
Scores of abnormal signals	−88.08	54.83
Scores of similar signals	−78.30	24.62
Scores of dissimilar signals	−49.11	29.41

testing dataset, and abnormal signals in the testing dataset (abbreviated as TraNS, TeNS, and TeAS) and present the corresponding distribution of each index. According to Figure 6, it can be seen that four indexes of TraNS and TeNS are extremely close, whereas four indexes of TraNS are different from those of TeAS. Therefore, we can exploit the mean and standard deviation of the indexes of TraNS to establish effective thresholds in the decision-making function so as to guarantee a satisfactory detection performance.

4.5. Effectiveness of Different Indexes. In this section, we explore the influence of different indexes on the performance of tool wear detection. For each individual index, we can calculate the corresponding value according to lines 11–14 of Algorithm 2 for anomaly detection. Furthermore, the performance of our MI-GAN is also presented as a comparison. In light of the results in Table 4, it can be observed that our MI-GAN has the best performance. The reason is that the proposed MI-GAN integrates multiple

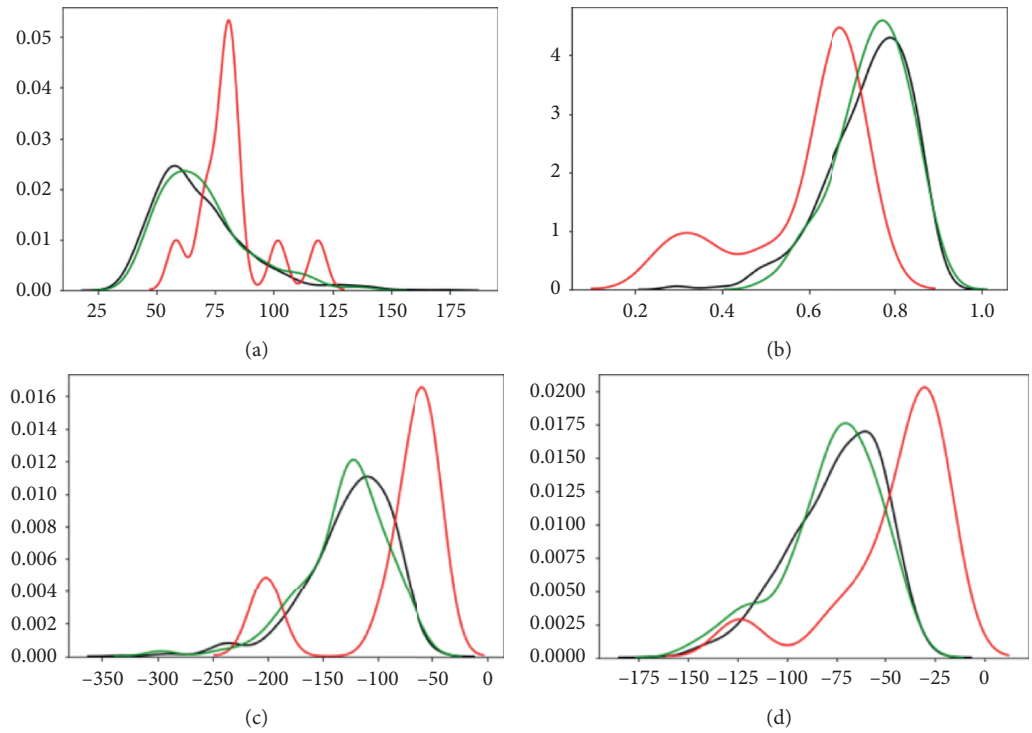


Figure 6: Distribution of four indexes: (a) the L_2-norm index between generated signals and actual signals; (b) the CORT index between generated signals and actual signals; (c) the fake-level score of actual signals; (d) the fake-level score of generated signals.

Table 4: The comparison results of different index settings.

Index	W-recall	W-precision	W-F
L_2-norm	0.81	0.83	0.82
CORT	0.83	0.85	0.84
Fake-level score of actual signals	0.87	0.91	0.88
Fake-level score of generated signals	0.83	0.88	0.85
MI-GAN	0.97	0.97	0.97

criteria and thus processes a more comprehensive detection capability.

4.6. Comparison with Other Machine Learning Methods. To further demonstrate the efficiency of the proposed MI-GAN, we compare it with other state-of-the-art machine learning methods: KNN [31], SVM [32], RF [33], CNN [14], Inception Net [34], DenseNet [35], ResNet [36], SE-Net [37], and Nonlocal-Net [38]. For a fair comparison, we reimplement those methods and adjust the hyperparameters on our training dataset to gain the best performance. The numerical results of different methods are illustrated in Table 5, from which we can conclude that the performance of CNN-based methods is better than that of the traditional machine learning ones (i.e., LNN, SVM, and RF). Besides, the proposed MI-GAN has a better performance with a significant improvement on accuracy, recall, precision, and F-measure compared with the CNN method. This is due to the fact that our MI-GAN could extract more critical features, and the excellent generation and discrimination capability of the MI-GAN could effectively handle the data imbalance challenge.

Table 5: The comparisons of different machine learning methods.

Method	W-recall	W-precision	W-F
KNN [31]	0.71	0.76	0.72
SVM [32]	0.67	0.75	0.69
RF [33]	0.69	0.76	0.71
CNN [14]	0.81	0.79	0.79
Inception Net [34]	0.85	0.89	0.83
DenseNet [35]	0.91	0.93	0.94
ResNet [36]	0.92	0.95	0.95
SE-Net [37]	0.93	0.96	0.95
Nonlocal-Net [38]	0.96	0.96	0.93
MI-GAN	0.97	0.97	0.97

5. Conclusion

In this paper, we have developed a novel MI-GAN to detect the tool wear conditions for imbalanced industrial data where normal signals are much larger than abnormal ones. TCN residual blocks have been utilized as the backbone of the generator and discriminator networks to leverage the ability of feature extraction. In addition, a decision-making function with multi-index has been exploited to aggregate more representative criteria such that the detection performance can be further improved. For the sake of verifying the effectiveness and feasibility of our MI-GAN, different experiments have been conducted on the real-time current signals sampling from CNC machines. The experimental

results have shown that our method indeed has an excellent performance on tool wear detection.

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the Key Project of Science and Technology Innovation 2030 supported by the Ministry of Science and Technology of China under Grant 2018AAA0101303, the National Natural Science Foundation of China under Grants 61803282 and 61973288, the National Key Research and Development Program of China under Grant 2018YFB2100801, the Fundamental Research Funds for the Central Universities of China under Grant 22120180561, the "Chenguang Program" Supported by the Shanghai Education Development Foundation and Shanghai Municipal Education Commission of China under Grant 18CG18, and the Shanghai Science and Technology Innovation Action Plan under Grants 20692193400 and 19511101300.

References

[1] P. Bhattacharyya, D. Sengupta, S. Mukhopadhyay, and A. B. Chattopadhyay, "Online tool wear prediction in milling operation using feed motor current signal," *International Journal of Production Research*, vol. 46, no. 4, pp. 1187–1201, 2006.

[2] S. Dutta, S. K. Pal, S. Mukhopadhyay, and R. Sen, "Application of digital image processing in tool condition monitoring: a review," *CIRP Journal of Manufacturing Science and Technology*, vol. 6, no. 3, pp. 212–232, 2013.

[3] P. Waydande, N. Ambhore, and S. Chinchanikar, "A review on tool wear monitoring system," *Journal of Mechanical Engineering and Automation*, vol. 6, no. 5A, pp. 49–53, 2016.

[4] Q. Hou, J. Sun, and P. Huang, "A novel algorithm for tool wear online inspection based on machine vision," *The International Journal of Advanced Manufacturing Technology*, vol. 101, no. 9–12, pp. 2415–2423, 2019.

[5] D. Kerr, J. Pengilley, and R. Garwood, "Assessment and visualisation of machine tool wear using computer vision," *The International Journal of Advanced Manufacturing Technology*, vol. 28, no. 7-8, pp. 781–791, 2006.

[6] P. Ong, W. K. Lee, and R. Lau, "Tool condition monitoring in CNC end milling using wavelet neural network based on machine vision," *The International Journal of Advanced Manufacturing Technology*, vol. 104, no. 1–4, pp. 1369–1379, 2019.

[7] M. Hassan, A. Damir, H. Attia, and V. Thomson, "Benchmarking of pattern recognition techniques for online tool wear detection," *Procedia CIRP*, vol. 72, pp. 1451–1456, 2018.

[8] D. Wu, C. Jennings, J. Terpenny, R. X. Gao, and S. Kumara, "A comparative study on machine learning algorithms for smart manufacturing: tool wear prediction using random forests," *Journal of Manufacturing Science and Engineering*, vol. 139, no. 7, p. 071018, 2017.

[9] Y. He, M. Huang, and W. Sun, "Tool wear status recognition based on Mahalanobis distance," *The Journal of Engineering*, vol. 2019, no. 23, pp. 8802–8805, 2019.

[10] B. Neef, J. Bartels, and S. Thiede, "Tool wear and surface quality monitoring using high frequency CNC machine tool current signature," in *Proceedings of the IEEE 16th International Conference on Industrial Informatics (INDIN)*, pp. 1045–1050, IEEE, July 2018, Porto, Portugal.

[11] G. Simon and R. Deivanathan, "Early detection of drilling tool wear by vibration data acquisition and classification," *Manufacturing Letters*, vol. 21, pp. 60–65, 2019.

[12] D. Wu, C. Jennings, J. Terpenny, and S. Kumara, "Cloud-based machine learning for predictive analytics: tool wear prediction in milling," in *Proceedings of the 2016 IEEE International Conference on Big Data (Big Data)*, pp. 2062–2069, IEEE, Washington, DC, USA, December 2016.

[13] C. Zhang and H. Zhang, "Modelling and prediction of tool wear using LS-SVM in milling operation," *International Journal of Computer Integrated Manufacturing*, vol. 29, no. 1, pp. 76–91, 2016.

[14] Y. Wang, W. Dai, and J. Xiao, "Detection for cutting tool wear based on convolution neural network," in *Proceedings of the 2018 12th International Conference on Reliability, Maintainability, and Safety (ICRMS)*, pp. 297–300, IEEE, Shanghai, China, October 2018.

[15] X.-C. Cao, B.-Q. Chen, B. Yao, and W.-P. He, "Combining translation-invariant wavelet frames and convolutional neural network for intelligent tool wear state identification," *Computers in Industry*, vol. 106, pp. 71–84, 2019.

[16] A. Kothuru, S. P. Nooka, and R. Liu, "Application of deep visualization in CNN-based tool condition monitoring for end milling," *Procedia Manufacturing*, vol. 34, pp. 995–1004, 2019.

[17] H. Zheng and J. Lin, "A deep learning approach for high speed machining tool wear monitoring," in *Proceedings of the 2019 3rd International Conference on Robotics and Automation Sciences (ICRAS)*, pp. 63–68, IEEE, Wuhan, China, June 2019.

[18] G. Martínez-Arellano, G. Terrazas, and S. Ratchev, "Tool wear classification using time series imaging and deep learning," *The International Journal of Advanced Manufacturing Technology*, vol. 104, no. 9–12, pp. 3647–3662, 2019.

[19] L. E. E. Ochoa, I. B. R. Quinde, J. P. C. Sumba, A. V. Guevara Jr., and R. Morales-Menendez, "New approach based on autoencoders to monitor the tool wear condition in HSM," *IFAC-PapersOnLine*, vol. 52, no. 11, pp. 206–211, 2019.

[20] X. Cao, B. Chen, B. Yao, and S. Zhuang, "An intelligent milling tool wear monitoring methodology based on convolutional neural network with derived wavelet frames coefficient," *Applied Sciences*, vol. 9, no. 18, p. 3912, 2019.

[21] W. Jiang, Y. Hong, B. Zhou, X. He, and C. Cheng, "A GAN-based anomaly detection approach for imbalanced industrial time series," *IEEE Access*, vol. 7, pp. 143608–143619, 2019.

[22] D. Li, D. Chen, J. Goh, and S. K. Ng, "Anomaly detection with generative adversarial networks for multivariate time series," 2018, https://arxiv.org/abs/1809.04758?.

[23] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," *Lecture Notes in Computer Science*, Springer, Cham, Switzerland, pp. 146–157, 2017.

[24] A. D. Chouakria and P. N. Nagabhushan, "Adaptive dissimilarity index for measuring time series proximity,"

Advances in Data Analysis and Classification, vol. 1, no. 1, pp. 5–21, 2007.

[25] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, https://arxiv.org/pdf/1803.01271.pdf.

[26] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," 2017, https://arxiv.org/abs/1701.07875.

[27] J. Wu, Z. Huang, J. Thoma, D. Acharya, and L. Van Gool, "Wasserstein divergence for GANS," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 653–668, Munich, Germany, September 2018.

[28] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," 2014, https://arxiv.org/abs/1412.6980.

[29] A. Creswell and A. A. Bharath, "Inverting the generator of a generative adversarial network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 7, pp. 1967–1974, 2018.

[30] S. Ruder, "An overview of gradient descent optimization algorithms," 2016, https://arxiv.org/abs/1609.04747.

[31] Z. Deng, X. Zhu, D. Cheng, M. Zong, and S. Zhang, "Efficient k NN classification algorithm for big data," *Neurocomputing*, vol. 195, pp. 143–148, 2016.

[32] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, vol. 9, no. 3, pp. 293–300, 1999.

[33] J. Cao and G. Fan, "Signal classification using random forest with kernels," in *Proceedings of the 2010 Sixth Advanced International Conference on Telecommunications*, pp. 191–195, IEEE, May 2010, Barcelona, Spain.

[34] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, Boston, MA, USA, June 2015.

[35] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4700–4708, Honolulu, HI, USA, July 2017.

[36] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, NV, USA, June 2016.

[37] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, Salt Lake City, UT, USA, June 2018.

[38] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7794–7803, Salt Lake City, UT, USA, June 2018.

Hindawi
Complexity
Volume 2020, Article ID 6431712, 16 pages
https://doi.org/10.1155/2020/6431712

Research Article

A Hybrid Prediction Method for Stock Price Using LSTM and Ensemble EMD

Yang Yujun,[1,2,3] **Yang Yimei**,[1,2] **and Xiao Jianhua**[1,2]

[1]*School of Computer Science and Engineering, Huaihua University, Huaihua 418008, China*
[2]*Key Laboratory of Intelligent Control Technology for Wuling-Mountain Ecological Agriculture in Hunan Province, Huaihua 418000, China*
[3]*Key Laboratory of Wuling-Mountain Health Big Data Intelligent Processing and Application in Hunan Province Universities, Huaihua 418000, China*

Correspondence should be addressed to Yang Yimei; yym1630@163.com

Received 24 July 2020; Revised 13 September 2020; Accepted 21 November 2020; Published 4 December 2020

Academic Editor: Cheng Lu

The stock market is a chaotic, complex, and dynamic financial market. The prediction of future stock prices is a concern and controversial research issue for researchers. More and more analysis and prediction methods are proposed by researchers. We proposed a hybrid method for the prediction of future stock prices using LSTM and ensemble EMD in this paper. We use comprehensive EMD to decompose the complex original stock price time series into several subsequences which are smoother, more regular and stable than the original time series. Then, we use the LSTM method to train and predict each subsequence. Finally, we obtained the prediction values of the original stock price time series by fused the prediction values of several subsequences. In the experiment, we selected five data to fully test the performance of the method. The comparison results with the other four prediction methods show that the predicted values show higher accuracy. The hybrid prediction method we proposed is effective and accurate in future stock price prediction. Hence, the hybrid prediction method has practical application and reference value.

1. Introduction

According to the statistics of China Securities Depository and Clearing Corporation, as of March 2020, there are 163.3 million securities investors in China. Stock price forecasting is a difficult and meaningful task for financial institutions and private investors. In order to effectively reduce investment risks and obtain stable returns on investment, many scholars put forward a large number of prediction models [1–9]. With the speedy development of big data application technology, especially the application of machine learning and deep learning in the financial field, it has a profound impact on investors. Research directions include low-frequency data and high-frequency data [10]. The previous research studies are mainly divided into two kinds of methods: fundamental analysis and technical analysis [11].

On the one hand, in technical analysis, people widely use mathematical statistical techniques to analyze historical stock price trends and predict recent stock prices. In recent years, many researchers have applied a variety of machine learning algorithms to analyze and predict stock prices, such as neural networks, multicore learning [12], stepwise regression analysis [13], and deep learning [14, 15]. Although many algorithms have achieved good results in certain aspects, there are many parameter configurations and data selection problems in the use of machine learning, which is still an important area of research. On the other hand, in the fundamental analysis [16–18], people mainly use natural language processing to analyze the company's financial news and financial statements to predict the future stock price trend.

The long-short term memory (LSTM) is a very good method in dealing with time series. Stock price data belongs

to time-series data. Therefore, many researchers use LSTM [19–26] to analyze and predict stock prices. Many studies have analyzed the correlation between time-series data [27–30], and the results show that LSTM has advantages in time-series data. In the literature [31], researchers used LSTM to predict the coding unit split, and the experimental results proved the advantages of LSTM in terms of efficiency. Time-series data trend research is also a new form of time-series data prediction, so LSTM is a natural choice.

Empirical mode decomposition (EMD) technology is usually applied to nonstationary and nonlinear signals. EMD can decompose nonlinear signals into several inherent modal functions (IMF) adaptively. EMD can effectively suppress continuous noise, such as Gaussian noise [32]. However, EMD cannot suppress intermittent noise and mixed noise. Ensemble empirical mode decomposition (EEMD) technology can solve the problem of noise-mode mixing [33]. In the EEMD algorithm, a group of white noise is first added to the original signal, and then it is decomposed into several IMF. The average value of the corresponding IMF set is regarded as the correct result. EEMD will separate the noise in different IMF from the original signal components [34], thus eliminating the noise-mode mixing phenomenon. In recent years, the application of EEMD has attracted the attention of many researchers and scholars [27, 35–44]. In order to solve the problem that noise in practical applications makes interference term retrieval difficult, Zhang et al. [35] proposed a technique based on EEMD and EMD to achieve automatic interference term retrieval from the spectral domain low-coherence interferometry. The proposed algorithm uses EEMD technology to make the relative error of coupling strength less than 2%. To solve the problem that Gaussian noise and non-Gaussian noise seriously hinder the detection of rolling bearing defects by traditional methods, Jiang et al. [36] proposed a new rolling bearing inspection method that combines bispectrum analysis with improved integrated EMD. This method uses ensemble empirical mode decomposition technology to have superior performance in reducing multiple background noises and can more effectively detect defects in rolling bearings. To solve the problem of the influence of the authenticity of the partial discharge signal on the evaluation accuracy of the transformer insulation performance, Wang et al. [37] proposed a method to suppress white noise in PD signals based on the integration of EMD and the combination of high-order statistics. This method uses EEMD decomposition to the threshold and reconstructs each IMF to suppress the white noise in each component. To solve the problem that most existing measurement methods only focus on mathematical values and are affected by measurement errors, interference, and uncertainty, Wei et al. [38] proposed a new time-history comparison for vehicle safety analysis by the integrating empirical mode decomposition method. This method uses EEMD decomposition to make the trend signal to reflect the overall change and is not affected by high-frequency interference. To solve the difficult problem of wind speed prediction, Yang and Yang [39] proposed a hybrid BRR-EEMD short-term prediction method for wind speed based on the EMD and Bayesian ridge regression (BRR). This hybrid method uses the Bayesian regression method and the EEMD to perform regression prediction on each subsequence decomposed by the EEMD and obtains good results in wind speed prediction. In order to find potential profit arbitrage opportunities when the returns of stock index futures contracts and stock index futures contracts continue to deviate from fair prices in irrational and nonefficiently operating markets, Sun and Sheng [40] proposed a time-series analysis method based on integrated EMD. This method uses EEMD to analyze the stock futures basis sequence and extracts a monotonically decreasing trend from the sequence to discover business opportunities. To improve the problem that a single method of predicting complex and nonlinear stock prices cannot achieve good results, Al-Hnaity and Abbod [41] proposed a hybrid integrated model based on ensemble empirical mode decomposition and backpropagation neural network to predict the closing price of the stock index. The researchers [41] have proposed five hybrid prediction models: ensemble EMD-NN, ensemble EMD-Bagging-NN, ensemble EMD-Crossvalidation-NN, ensemble EMD-CV-Bagging-NN, and ensemble EMD-NN. The experimental results show that the performance of the ensemble EMD-CV-Bagging-NN, ensemble EMD-Crossvalidation-NN, and ensemble EMD-Bagging-NN models based on ensemble EMD are all a grade higher than that of the ensemble EMD-NN model and significantly higher than the single neural network model.

The typical forecasting scheme is based on the forecast of the time-series data itself and does not deal with the time-series data itself. It has become a challenge that how to combine the existing forecasting methods to improve the forecasting effect by decomposing the time-series data. The above methods use EEMD to decompose the time-series data to improve the performance of the algorithm. How to effectively decompose complex and nonlinear stock time-series data for prediction has been puzzled by many researchers. Due to the uncertainty and nonlinearity of the stock time series, the deviation of a single method to predict stock prices is generally relatively large. The abovementioned hybrid method does indeed improve the algorithm significantly. Therefore, it can be boldly guessed that the hybrid method generally can get better prediction results than the single specific method. Besides, the original complex time series was decomposed by the EEMD method into several relatively stable subsequence time series. By effectively combining several current effective forecasting methods, the forecasting results of relatively stable subsequence time series are theoretically better. Combining the features of the EEMD method based on the improved empirical mode decomposition method and the LSTM machine learning algorithm, this paper proposes a hybrid LSTM-EEMD method for stock index price prediction.

The rest of this paper is organized as follows. Three related terminology such as EMD, EEMD, and BRR are presented in Section 2, while Section 3 briefly introduces the flowchart of our proposed LSTM-EEMD method and the structure of our proposed hybrid LSTM-EEMD method. In Section 4, we describe the experiment data collection,

experiment data preprocessing, and modeling processing. In Section 5, we describe the experimental results of our proposed hybrid LSTM-EEMD method and analyze the results of simulation experiment of our proposed hybrid LSTM-EEMD method for prediction. Finally, the conclusion of this paper and some future works are described in Section 8.

2. Related Works

Over the years, many studies in the financial field have focused on the problems of stock price prediction. These studies mainly focus on three important research directions: (1) based on the machine learning method; (2) based on the time-series analysis method; and (3) based on the hybrid method. Below, we first briefly introduce related terminology. Then, we introduce the LSTM and EEMD related to this study.

2.1. Stock Price Predicting. Stock prices are a highly volatile time series. Stock prices are affected by various factors such as national policies, interest rates, exchange rates, industry news, inflation, monetary policies, temporary events, investor sentiment, and human intervention. Predicting stock prices on the surface requires the establishment of a model of the relationship between stock prices and these factors. Although these factors will temporarily change the stock price, in essence, these factors will be reflected in the stock price and will not change the long-term trend of the stock price. Therefore, stock prices can be predicted simply with historical data.

This paper believes that there are many studies using a single analysis method to predict stock market trends, but the results are not good. Need to consider a variety of factors or use a variety of techniques to build a hybrid model to further explore the prediction of stock prices. In-depth systematic research is required to answer the following research questions:

RQ1 Which factors or combinations of factors most affect the trend of the stock?

RQ2 What kind of analysis technology combination is most suitable for stock trend prediction?

RQ3 Do we need to use deep learning methods to mine data in order to better discover the internal relationship between the stock market and influencing factors?

RQ4 Whether the predictability of the analysis model depends on specific stock company characteristics, such as the domain, shareholder background, and policies?

RQ5 In the context of stock market forecasting, have we developed some effective forecasting methods?

RQ6 We should focus on the analysis of the specific nature of the stock price itself, rather than solving general relationship problems. Whether the price analysis driven by influencing factors can be more effective?

2.2. LSTM. The long-short term memory neural network is generally called LSTM. The LSTM was proposed by Hochreiter and Schmidhuber [27] in 1997. The LSTM is a special type of recurrent neural network (RNN). The biggest feature of the RNN which was improved and promoted by Alex Graves is that long-term dependent information of data can be obtained. LSTM has been widely used in many fields and has achieved considerable success in many problems. Since LSTM can remember the long-term information of data, the design of LSTM can avoid the problem of long-term dependence. Currently, the LSTM is a very popular time-series forecasting model. Below, we first introduce the RNN network, followed by the LSTM neural network.

3. Preliminaries

3.1. Recurrent Neural Network. When we deal with problems related to the timeline of events, such as speech recognition, sequence data, machine translation, and natural language processing, traditional neural networks are powerless. RNN is specifically proposed to solve these problems. Because the correlation between the contexts of the text needs to be considered in the word processing, the weather conditions of consecutive days and the relationship between the weather conditions of the day and the past days need to be considered when predicting the weather.

The RNN has a chain form of repeating neural network modules. In the standard RNN, this repeated structural module has only a very simple structure, such as a tanh layer. The simple structure of the recurrent neural network is shown in Figure 1.

The design intent of RNN is to solve nonlinear problems with timelines. The way of internal connection of the recurrent neural network generally only feeds forward the data, but in the bidirectional recurrent neural network, it allows the forward and backward directions to feedback the data. The RNN has designed a feedback mechanism, so the RNN can easily update the weight or residual value of the previous step. The design of the feedback mechanism is very suitable for time-series forecasting. The RNN can extract rules from historical data and then use the rules to predict time series. Figure 1 shows the simple structure of the RNN, and Figure 2 shows the expanded diagram of the basic structure of the RNN. The left side of the arrow with unfold label in Figure 2 is the basic structure of the recurrent neural network. The right side of the arrow with unfold label in Figure 2 is a continuous 3-level expansion of the basic structure of the recurrent neural network. An input data x_t is input into module h of the RNN. The y_t is an output of module h of the RNN values at time t. Like other neural networks, the recurrent neural network shares all parameters of each layer, such as W_{hx}, W_{hh}, and W_{yh} in Figure 2. As shown in Figure 2, the RNN shares two input parameters W_{hx} and W_{hh}, and one output parameter W_{yh}. As we all know, the number of parameters in each layer of a general multilayer neural network is different. Looking at Figure 2, we feel that the operation of each step of the recurrent neural network is the same on the surface. In fact, the output y_t and the input x_t are different. The number of parameters of the

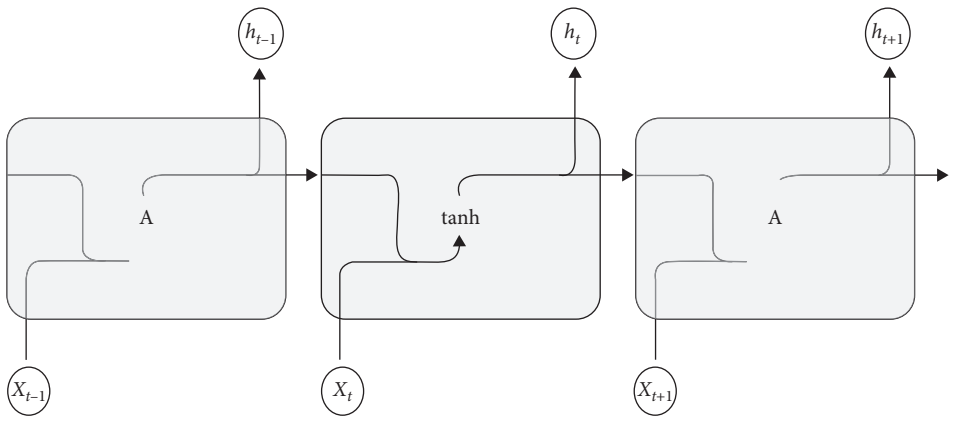


Figure 1: The simple structure of the RNN.

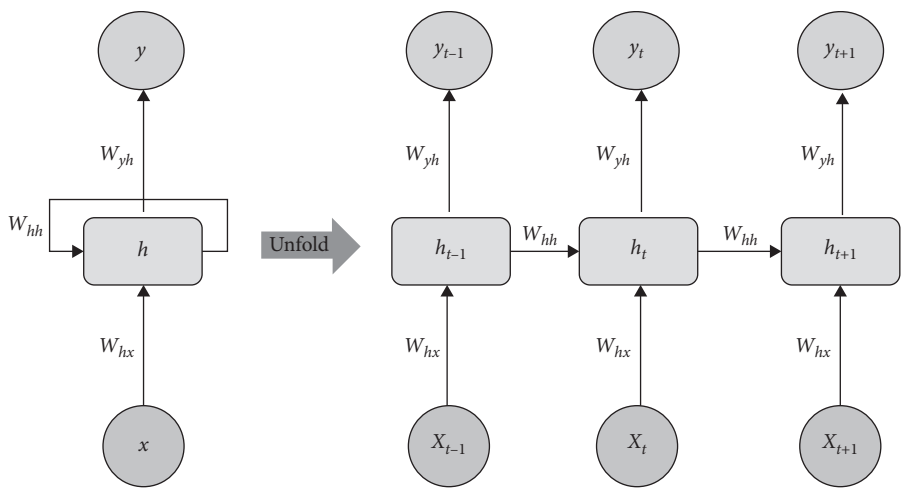


Figure 2: The expanded diagram of the basic structure of the RNN.

recurrent neural network will be significantly reduced during training. After multilevel expansion, the recurrent neural network becomes a multilayer neural network. Looking closely at Figure 2, we find that W_{hh} between layer $h_{t\text{-}1}$ and h_t is the same as W_{hh} between layer h_t and h_{t+1} in form. In value and meaning, W_{hh} between layer $h_{t\text{-}1}$ and h_t is different as W_{hh} between layer h_t and h_{t+1}. Similarly, W_{hx} and W_{yh} have similar situations.

Although each layer of the RNN neural network has output and input modules, the output and input modules of some layers can be omitted in specific application scenarios. For example, in language translation, we only need the overall language symbol output after the last language symbol is input and do not need to know the language symbol output after each language symbol is input. The main feature of RNN is self-expanding, with multiple hidden layers.

As we all know, during network training, the recurrent neural network models are prone to disappearing gradients. Once the gradient of the model disappears completely, the algorithm enters an endless loop, and the network training will not end, eventually leading to RNN paralysis. Therefore, simple RNNs are prone to gradient disappearance problems and are not suitable for long-term predicting.

The purpose of designing LSTM is to avoid or reduce the appearance of the problem of vanishing gradient while dealing with long-term correlation time series by simple RNN. Based on the simple RNN, the LSTM adds the output gates, the input gates, and the forget gates. In Figure 3, all three gates are replaced by σ, which can effectively prevent the gradient from being eliminated. Therefore, LSTM can solve long-term dependence problems. The purpose of designing memory neurons is to store some LSTM important information for state information. In addition, in general, each gate has an activation function. This function performs nonlinear transformations or trade-offs on data. Generally, the forget gate f_t can filter some status information. Equations (1)–(6)fd6 associated with the LSTM neural network is shown below. They are for the forget gate, input gate, inverse of the memory cell, memory cell, output gate, and output, respectively:

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

$$i_t = \sigma\left(W_i \cdot \left(h_{t-1} \| x_t\right) + b_i\right), \tag{2}$$

$$\widetilde{C}_t = \tanh\left(W_C \cdot \left(h_{t-1} \| x_t\right) + b_C\right), \tag{3}$$

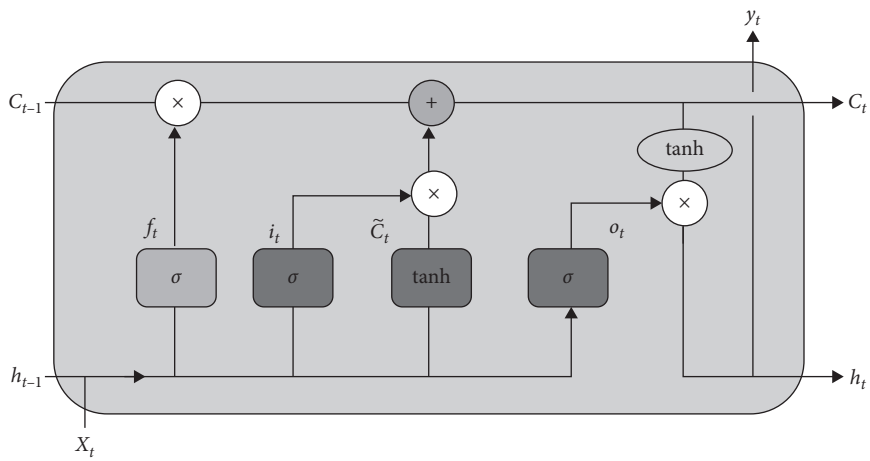


Figure 3: The basic structure of the LSTM neural network.

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

$$o_t = \sigma\left(W_o \cdot \left(h_{t-1} \| x_t\right) + b_o\right), \tag{5}$$

$$h_t = o_t \times \tanh\left(C_t\right). \tag{6}$$

3.2. EMD. The empirical mode decomposition (EMD) proposed by Huang et al. in 1998 [42] is a widely used adaptive time-frequency analysis method. Empirical mode decomposition is an effective decomposition method for time-series data. Due to the common local features of time-series data, the EMD method has extracted the required data from them and obtained very good results in applications fields. Hence, many scholars apply the EMD method in many fields successfully. The prerequisite for EMD decomposition is the existence of the following three assumptions [43]:

The following briefly introduces the decomposition process:

(1) Suppose there is a signal $s(i)$ with the black line in Figure 4. The extreme value of the signal is shown in red and blue dots. Form the upper wrapping line through all the blue dots and form the lower wrapping line through all the red dots.

(2) Calculate the average value of the lower and upper wrapping lines to form an average purple-red line $m(i)$. Here, define the discrepancy line $d(i)$ as

$$d(i) = s(i) - m(i). \tag{7}$$

(3) Judge whether the discrepancy line $d(i)$ is an IMF according to IMF judgment rules. If the discrepancy line $d(i)$ comply with IMF judgment rules, the discrepancy line $d(i)$ is the ith IMF $f(i)$. Otherwise, the discrepancy line $d(i)$ is considered the signal $s(i)$ and repeat two steps above until $d(i)$ complies with IMF judgment rules. After this, the IMF $f(t)$ is defined as

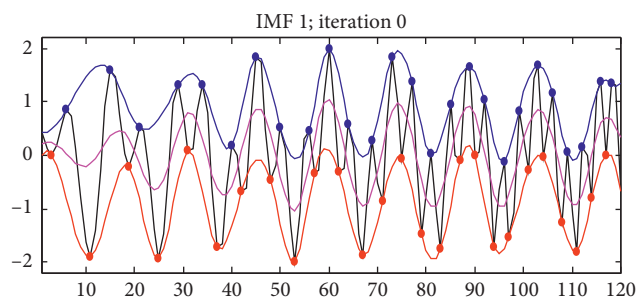


Figure 4: The decomposition process chart of sequences.

$$f(t) = d(t), \quad t = 1, 2, 3, \ldots, n-1. \tag{8}$$

(4) Calculate and get the IMF $f(t)$ by

$$r(t) = s(i) - c(t), \tag{9}$$

where $r(t)$ is considered the residual signal.

(5) Repeat execution the four steps above N times until running status meets stop conditions. Obtain the N IMFs which meet with

$$\begin{cases} r_1 - c_2 = r_2, \\ \vdots \\ r_{N-1} - c_N = r_N. \end{cases} \tag{10}$$

(1) Next level signal has to contain more than two extreme values: one is the minimum value and the other is the maximum value

(2) Determine the time scale of signal characteristic based on the time difference between two extreme values

(3) If the data has only an inflection point and no extremum, more times judgments are needed to reveal the extremum

Finally, the following equation (11) expresses the composition of the original signal $s(i)$:

$$s(i) = \sum_{j=1}^{N} f_j(t) + r_j(t). \tag{11}$$

3.3. EEMD. A classical EMD has mode mixing problems when decomposing complex vibration signals. To solve the above problem, Wu and Huang [44] proposed the EEMD method in 2009. The EEMD method is short for ensemble empirical mode decomposition method. EEMD is commonly used for nonstationary signal decomposition. However, the EEMD method is significantly different from WT transform and FFT transform. Here, WT is the wavelet transform and FFT is the fast Fourier transform. Without the need for basis functions, the EEMD method can decompose any complex signal. At the same time, the EEMD method can decompose any signal into many IMFs. Here, IMF is the intrinsic modal function. The decomposed IMF components contain local different feature signals. EEMD can decompose nonstationary data into multiple stable subdata and then use Hilbert transform to get the time spectrum, which has important physical significance. Comparative analysis with FFT and WT decomposition, the EEMD has the characteristics of intuitive, direct, posterior, and adaptive. The EEMD method has adaptive characteristics because of the local features of the time-series signal. The following briefly introduces the process of EEMD decomposing data.

Assume that the EEMD will decompose the sequence X. According to the steps of EEMD decomposition, n subsequences will be obtained after decomposition. These n subsequences include $n-1$ IMFs and one remaining subsequence R_n. Here, these $n-1$ IMFs are $n-1$ component subsequence $C_i(i = 1, 2, ..., n-1)$ of the original sequence X. These $n-1$ IMFs are named $\mathrm{IMF}_i(i = 1, 2, ..., n-1)$. The remaining subsequence R_n is sometimes named residual subsequence. The detailed steps of using EEMD to decompose the sequence are introduced as follows:

(1) Suppose there is a signal $s(i)$ with the black line in Figure 4. The extreme value of the signal is shown in red and blue dots. Form the upper wrapping line through all the blue dots and form the lower wrapping line through all the red dots. The lower wrapping line is shown in the red line in Figure 4. And the upper wrapping line is the blue line in Figure 4.

(2) Calculate the average value of the lower and upper wrapping lines to form a mean line $m(i)$ which is shown in purple-red line in Figure 4.

(3) Obtain the first component IMF_1 of the signal $s(i)$. The IMF_1 is obtained by calculation formula $d(i) = s(i) - m(i)$. This formula means the difference of the original signal $s(i)$ minus the mean line $m(i)$ of the lower and upper wrapping lines.

(4) Take the first component IMF_1 $d(i)$ as a new signal $s(i)$ and repeat execution the three steps above until running status meets stop conditions. The following describes the stop condition in detail:

(a) The average $m(i)$ is approximately 0

(b) The number of signal lines $d(i)$ passing through zero points is greater than or equal to the number of extreme points

(c) The number of iterations reaches the set maximum

(5) Take subsequence $d(i)$ as the ith IMF fi ($i = 1, 2, \ldots, n-1$). Obtain the residual R by calculating the formula $r(i) = s(i) - d(i)$.

(6) Take the residual $r(i)$ as the new signal $s(i)$ to calculate the $(i+1)$th IMF, and repeat execution the five steps above until running status meets stop conditions. The following describes the stop condition in detail:

(a) The signal $s(i)$ has been completely decomposed and has obtained all IMF

(b) The number of decomposition level reaches the set maximum

Finally, the following equation (12) expresses the composition of the original sequences and n subsequences decomposed by the EEMD:

$$x(t) = \sum_{i=1}^{n-1} (C_i) + R_n, \quad i = 1, 2, \ldots, n-1, \tag{12}$$

where the number n of subsequences depends on the complexity of the original sequences. Figure 5 shows the sin time series represented by equation (13) and the IMF diagram of it which is decomposed by the EEMD:

$$x(t) = \sin(20\pi t) + 2^* \sin(200\pi t) + 3t, \tag{13}$$

where $t = 0, f, 2f, \ldots, 1000f, f = 0.001$.

4. Methodology

The principle of our hybrid prediction method LSTM-EEMD for stock price based on LSTM and EEMD is introduced in detail in this section. These theories are the theoretical formation of our forecasting methods. The following first introduces the flowchart, the basic structure, and the process of the LSTM-EEMD hybrid stock index prediction method based on the ensemble empirical mode decomposition and the long-short term memory neural network. Our proposed hybrid LSTM-EEMD prediction method first uses the EEMD to decompose the stock index sequences into a few simple stable subsequences. Then, the predict result of each subsequence is predicted by the LSTM

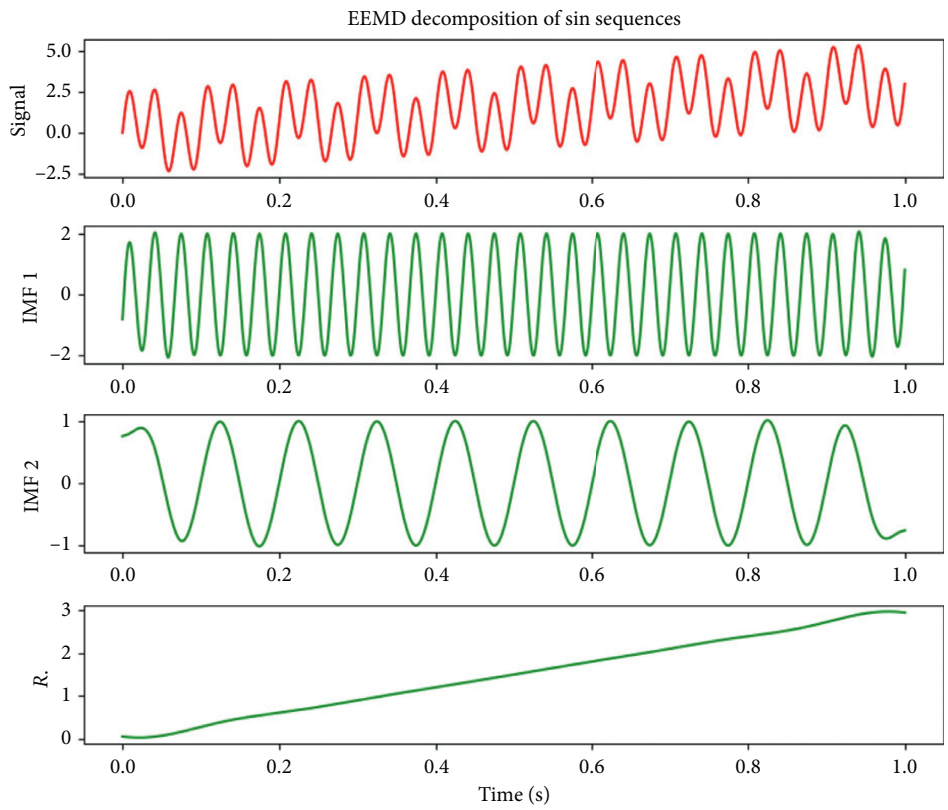


Figure 5: The sequences of sin signal decomposed by the EEMD.

method. Finally, the LSTM-EEMD obtains the final prediction result of the original stock index sequence by fusing all LSTM prediction results of several stock index subsequences.

Figure 6 shows the structure and process of the LSTM-EEMD method. The basic structure and process of the LSTM-EEMD predict method include three modules. The three modules are the EEMD decomposition module, LSTM prediction module, and fusion module. Our proposed LSTM-EEMD prediction method includes three stages and eight steps. Figure 7 shows three stages and eight steps of our proposed method. The three stages of the proposed hybrid LSTM-EEMD prediction method are input data, model predict, and evaluate model. The model evaluation stage and the data input stage each include 4 steps, and the model prediction stage includes 3 steps. The hybrid LSTM-EEMD prediction method is introduced in detail as follows:

(1) The simulation data is generated. And the real-world stock index time-series data are collected. Then, the original stock index time-series data are preprocessed to make the data format of stock index time series satisfy the format requirements for decomposition of the EEMD. Finally, the input data X of the LSTM-EEMD hybrid prediction method is formed.

(2) The input data X is decomposed into a few sequences by the EEMD method. If n subsequences are obtained, then there are one residual subsequence R_n and $n-1$ subsequences. These n subsequences are expressed as R_n, IMF_1, IMF_2, IMF_3, ..., IMF_{n-1}, respectively.

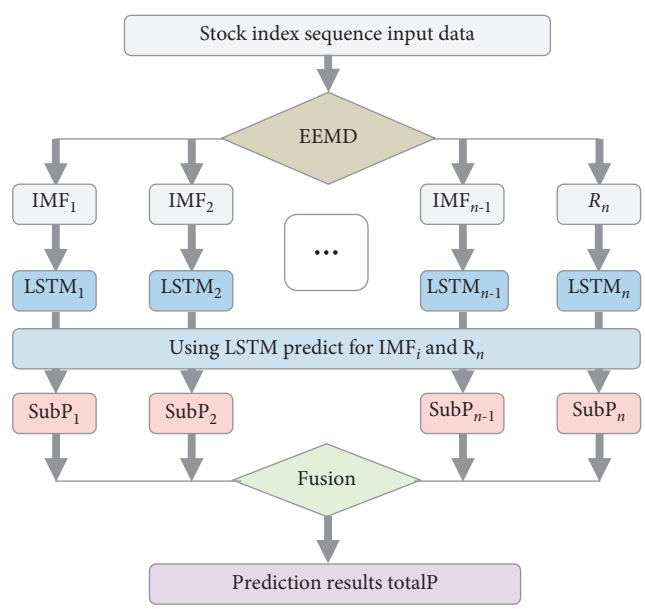


Figure 6: The structure and process of the LSTM-EEMD method.

(3) The prediction process of any subsequence is not affected by the prediction process of other subsequences. A LSTM model is established and trained for each subsequence. Hence, we need to build and

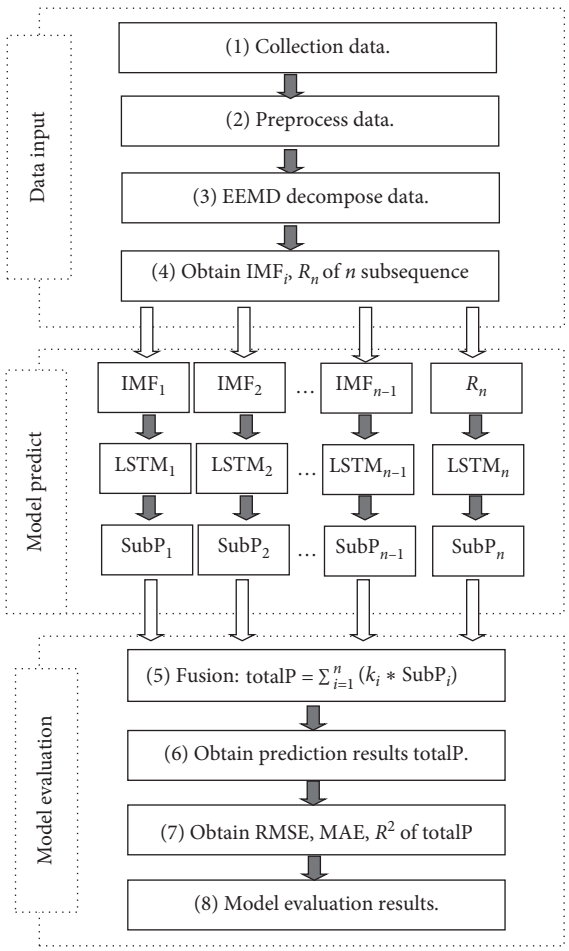


Figure 7: The data flowchart of our proposed hybrid method.

train n LSTM for n independent subsequences. These n independent LSTM models are named $\text{LSTM}_k(k = 1, 2, \ldots, n-1, n)$, respectively. We use the n LSTM models to predict these n independent subsequences and get n prediction values of the stock index time series. These n prediction values are named SubP_k $(k = 1, 2, \ldots, n-1, n)$, respectively.

(4) Fusion function is the core of hybrid method. At present, there are many fusion functions, such as sum, weighted sum, weighted product, and so on. The function of these fusion functions is to merge several results into the final result. In this paper, the proposed hybrid stock prediction method selected the weighted sum as the fusion function. The weighted results of all subsequences are accumulated to form the final prediction result for the original stock index data. The weight here can be preset according to the actual application. In this paper, we use the same weight of each subsequence and the weight of each subsequence is 1.

(5) Finally, we compare the predicted values with the actual value of stock index time-series data sequence and calculate the values of RMSE, MAE, and R^2. We use three evaluation criteria of the RMSE, MAE, and R^2 to evaluate the LSTM-EEMD hybrid prediction method. According to these evaluation values, the pros and cons of the method can be judged.

Figure 7 shows the predict progress and data flowchart of the proposed LSTM-EEMD method in this paper. The predict progress in Figure 7 can be introduced in 3 stages. The three stages are input data, model predict, and evaluate model. The stage of input data is divided into 4 steps. The four steps are collect data, preprocess data, decompose data by the EEMD, and generate n sequence. There are n LSTM model in the stage of model predict. The input data of the LSTM model is n sequences generated in the previous stage. These LSTM models separately predict n sequences to obtain n prediction results. The stage of the evaluate model is also divided into 4 steps. The first step is to fuse n predicted values with weights. In this paper, we choose weighted addition as the fusion function. The weighted addition fusion function sums the n prediction value with certain weights. The output result of the weighted addition fusion function is the prediction result of the stock index time-series data. Finally, we need to calculate the value of R^2, MAE, and RMSE of the prediction results before evaluating the proposed hybrid prediction model. The quality of the proposed hybrid prediction model can directly evaluated by the values of R^2, MAE, and RMSE.

5. Experiment Data

The experiment data in our research of this paper is introduced in detail in this part. We selected two types of experiment data to test in this paper in order to better evaluate the prediction effect of this method. The first type of experiment data is artificially simulated data generated automatically by computer. The correctness and effectiveness of our method are verified by these artificially simulated data. The second type of experimental data is real-world stock index data. Only the actual data of the society can really test the quality of the method. The model tested through social actual data is the most fundamental requirement for applying our proposed method to some real-world fields.

5.1. Artificially Simulated Experimental Data. We use artificially simulated experiment data to verify the effectiveness and correctness of our method. To get effective and accurate experiment result, the artificially generated simulation experiment data should be long enough. Hence, in the experiment, we choose the length of the artificially simulated experiment data to be 10,000. The artificially simulated experiment data is created by the computer according to the sin function of formula (13).

5.2. Real-World Experimental Data. In order to empirically study the effect of the proposed prediction method, we collected stock indices data in the real-world stock field as experiment data from Yahoo Finance. To obtain more objective experimental results, we choose 4 stock indices from

different countries or regions. These stock indices are all very important stock indices in the world's financial field.

The four stock indices in this experiment are ASX, DAX, HSI, and SP500. The SP500 is used as an abbreviation of the Standard&Poor's 500 in this paper. The SP500 is a US stock market index. This stock index is a synthesis of the stock indexes of 500 companies listed on NASDAQ and NYSE. The HSI is used as an abbreviation of the Hang Seng Index. The HSI is an important stock index for Hong Kong in China. The German DAX is used as an abbreviation of the Deutscher Aktienindex in Germany in Europe. The DAX stock index is compiled by the Frankfurt Stock Exchange. DAX company consists of 30 major German companies, so it is a blue chip market index. The ASX is used as an abbreviation of Australian Securities Exchange and is a stock index compiled by Australia.

The datasets of every stock index include five daily properties. These five properties are transaction volume, highest price, lowest price, closing price, and opening price. The data time interval of each stock index is from the opening date of the stock to May 18, 2020.

5.3. Data Preprocessing. In order to obtain good experiment results, we try our best to deal with all the wrong or incorrect data in the experiment data. Of course, more experimental data is also an important condition for obtaining good experimental results. Incorrect data mainly include records with zero trading volume and records with exactly the same data for two and more consecutive days. After removing wrong or incorrect noise data from original data, we show the trend of the close price for the four major stock indexes in Figure 8.

In the experiment, we usually first standardize the experimental data. Let $X_i = \{x_i(t)\}$ be the ith stock time-series index at time t, where $t = 1, 2, 3, \ldots, T$ and $i = 1, 2, 3, \ldots, N$. We define the daily return logarithmic as shown in the formula $G_i(t) = \log(x_i(t)) - \log(x_i(t-1))$. We define the daily return standardization as shown in the formula $R_i(t) = (G_i(t) - \langle G_i(t)\rangle)/\delta$, where $\langle G_i(t)\rangle$ is the mean values of the daily return logarithmic $G_i(t)$ and δ is the standard deviation of the daily return logarithmic $G_i(t)$:

$$\delta = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2}, \tag{14}$$

where $\overline{x} = 1/n \sum_{i=1}^{n} x_i$.

6. Experiment Results of Other Methods

In order to compare the performance of other forecasting methods, we comparatively studied and analyzed the results of other three forecasting methods on the same data in the experiment. Table 1 shows the experiment results of five prediction methods. Since the LSTM is introduced above, it will not be repeated here. Firstly, SVR, BARDR, and KNR are briefly introduced. Then, the experimental results of these three methods are analyzed in detail.

6.1. SVR. SVR is used as an abbreviation of the Support Vector Regression. The SVR is a widely used regression method. Refer to the manual of the *libsvm* toolbox, the loss, and penalty function control the training process of machine learning. The *libsvm* toolbox is support vector machine toolbox software, and this toolbox is mainly used for SVM pattern recognition and regression software package. In the experiment, the SVR used was a linear kernel, which was implemented with *liblinear* instead of *libsvm*. The SVR should be extended to large number of samples. The SVR can choose a variety of penalty functions or loss functions. SVR has 10 parameters, and the settings of these parameters are shown in Table 2.

6.2. BARDR. BARDR is short for Bayesian ARD Regression which is used to fit the weight of the regression model. This method assumes that the weight of the regression model conforms to the Gaussian distribution. To better fit the regression model weights, the BARDR uses the ARD prior technique. We assume the distribution of the regression model weights conforms to the Gaussian distribution. The parameter alpha and parameter lambda of BARDR are the precision of the noise distribution and the precisions of the weights distributions, respectively. BARDR has 12 parameters. Table 3 shows the settings of these parameters.

6.3. KNR. KNR is used as an abbreviation of the K-nearest Neighbors Regression. The K-nearest Neighbors Regression model is also a parameterless model. The K-nearest Neighbors Regression just uses the target value of the K-nearest training samples to make a decision on the regression value of the sample to be tested. That is, predict the regression value based on the similarity of the sample. Table 4 shows the KNR 8 parameters and the settings of these parameters.

The experimental results of other four methods and our proposed two methods for five sequences of sin, SP500, HSI, DAX, and ASX are shown in Table 1. In the experiment, we preprocessed the five sequences for those methods. Table 5 shows the length of the experimental data. Table 1 shows the experiment values of R^2 (R Squared), MAE (Mean Absolute Error), and RMSE (Root Mean Square Error). According to the real results and the predicted results, we try to get smaller experimental value of MAE, RMSE, and R^2. The smaller the result of MAE or RMSE is, the better experimental values of the method are. However, the larger the result of R^2 is, the better the prediction effect of the method is.

For comparison, we show the top two results of each sequences in bold, as shown in Table 1. Among the above four traditional methods (SVR, BARDR, KNR, and LSTM) for predicting sin artificial sequences data, BARDR and LSTM are found to be the best methods. When predicting SP500, HSI, and DAX real time-series data, the BARDR method and LSTM method have the best results. However, the BARDR method and SVR method have the best results while predicting ASX real time-series data. Therefore, among the above four traditional methods, the BARDR

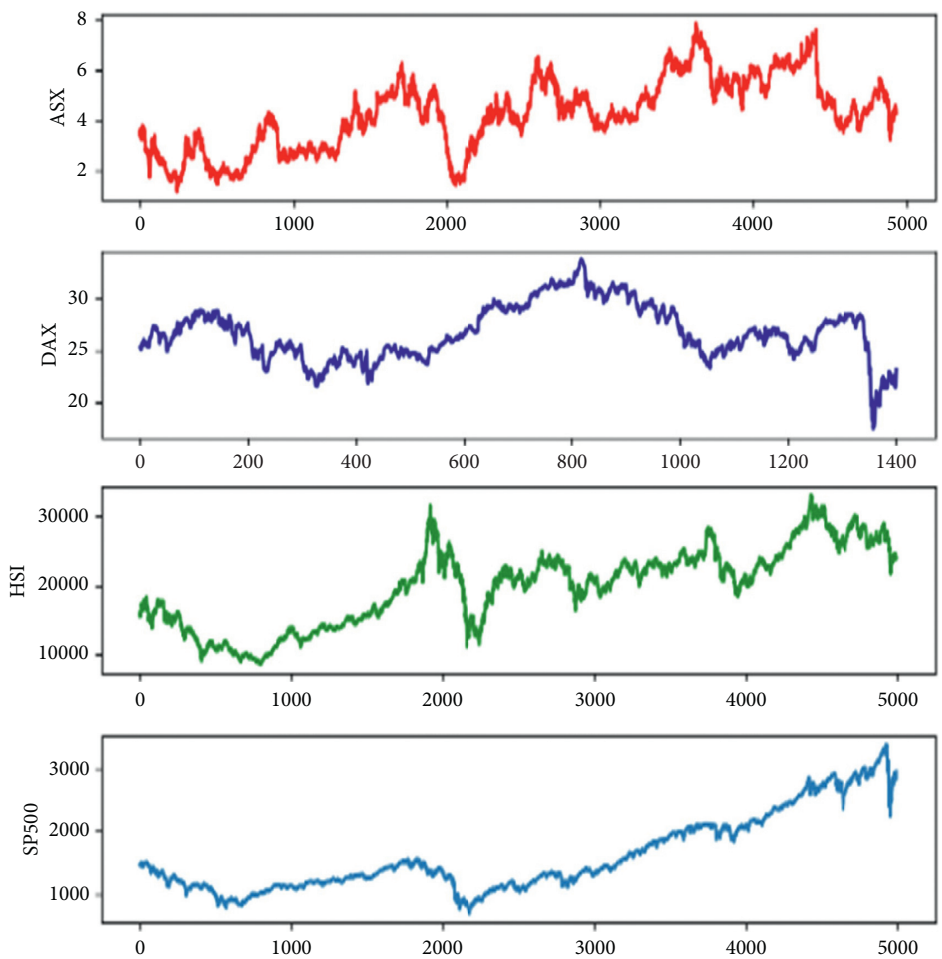


Figure 8: The trend of the four major stock indexes.

Table 1: Prediction results of time series.

Method		Sin	SP500	HSI	DAX	ASX
SVR	RMSE	0.282409	1219.904783	1334.991215	0.653453	0.206921
	MAE	0.238586	990.639020	1282.862498	0.415796	0.146022
	R^2	0.968293	−1.665447	−1.088731	0.938262	0.959223
BARDR	RMSE	0.012349	114.140660	326.487107	0.387025	0.110975
	MAE	0.011054	110.146472	233.641980	0.256125	0.075355
	R^2	0.999939	0.998275	0.992213	0.978343	0.988271
KNR	RMSE	0.567807	1198.484251	553.641943	0.740763	0.303352
	MAE	0.443162	937.309021	452.300293	0.445212	0.186848
	R^2	0.871824	−1.572662	−1.239516	0.920661	0.912361
LSTM	RMSE	**0.059597**	**30.261563**	306.481549	0.654662	0.135236
	MAE	**0.051376**	**19.164419**	224.250031	0.504645	0.106639
	R^2	**0.998562**	**0.997128**	0.980270	0.962750	0.931165
Proposed LSTM-EMD	RMSE	**0.032478**	76.562082	**78.487994**	**0.403972**	**0.066237**
	MAE	**0.026175**	47.317009	**71.062834**	**0.217733**	**0.045643**
	R^2	**0.999569**	0.985523	**0.987594**	**0.979444**	**0.996174**
Proposed LSTM-EEMD	RMSE	0.201358	**48.878331**	**48.783906**	**0.324303**	**0.101511**
	MAE	0.164348	**38.556500**	**38.564812**	**0.247722**	**0.081644**
	R^2	0.982980	**0.994171**	**0.988451**	**0.986832**	**0.991071**

Table 2: The value of eight parameters of SVR.

Parameter name	Parameter type	Parameter value
epsilon	Float	0.0
Tol	Float	$1e-4$
C	Float	1.0
Loss	"epsilon" or "squared_epsilon"	Epsilon
fit_intercept	Bool	True
intercept_scaling	Float	1.0
Dual	Bool	True
Verbose	Int	1
random_state	Int	0
max_iter	Int	1000

Table 3: The value of eight parameters of BARDR.

Parameter name	Parameter type	Parameter value
n_iter	Int	300
Tol	Float	$1e-3$
alpha_1	Float	$1e-6$
alpha_2	Float	$1e-6$
lambda_1	Float	$1e-6$
lambda_2	Float	$1e-6$
Compute_score	Bool	False
Threshold_lambda	Float	10000
fit_intercept	Bool	True
Normalize	Bool	False
copy_X	Bool	True
Verbose	Bool	False

method and LSTM method are the best methods for the five sequences data.

Carefully observed Table 1, we found that the remaining five methods, in addition to the KNR, all have very good prediction effects on sin sequences. The R^2 evaluation indexes of SVR, BARDR, LSTM, LSTM-EMD, and LSTM-EEMD methods are all greater than 0.96. BARDR, LSTM, and LSTM-EMD have the best prediction effect, and their R^2 evaluation indexes are all greater than 0.99. Among them, BARDR has the best prediction effect, and its R^2 evaluation value is greater than 0.9999. These values show that the method has better prediction effect for the time series of change regularity and stability, especially the BARDR method. Here, the result of choosing one of the six methods shows the prediction effect in Figure 9. To make the resulting map clear and legible, the resulting graph of Figure 9 displays only the last 90 prediction results.

Observing the SVR experiment results in Table 1, we found that the prediction values of this method on DAX, ASX, and sin is better than that of SP500 and HSI time-series data. Figure 10 shows the prediction results of the SVR on ASX, which has a better prediction effect on ASX stock data. Because the change of sin sequence has good regularity and stability, the SVR method is more suitable to predict sequence with good regularity and stability. It can be speculated that the DAX and ASX time-series data have good regularity and stability. However, the SVR method predicts SP500 and HSI sequences data. The prediction effect is very poor. This shows that SP500 and HSI sequences data changes have poor regularity and stability.

Observing the experiment results of the KNR in Table 1, we found this method has similar performance to the SVR method. The prediction results on DAX, ASX, and sin sequence are better than that on SP500 and HSI sequence. Especially for stock SP500 and HSI sequence, the prediction effect is poor. The stock time series is greatly affected by people activities, so the changes in stock sequence are complicated, irregular, and unstable. It can be inferred that the KNR is not suitable to predict the stock sequence, so the KNR is not suitable to predict sequence predictions with unstable and irregular changes.

In Table 1, observing the experiment results of the BARDR and LSTM, we found the performance of the BARDR and LSTM methods is relatively good. These two methods not only have better prediction effects on DAX, ASX, and sin time-series data but also have more significant prediction effects on SP500 and HSI sequence. In particular, the prediction values of stock SP500 and HSI sequences are much better than that of KNR and SVR methods. The changes of stock sequence data are irregular, complex, and unstable, but the prediction effects of the BARDR and LSTM are still very good. It can be concluded that the BARDR and LSTM methods can predict stock time series, so the BARDR and LSTM are more suitable to predict sequence predictions with unstable and irregular changes. Figure 11 shows the prediction results of the BARDR for DAX time-series data, which has a better prediction effect on DAX stock time-series data.

In Table 1, observing the experiment results of the BARDR and LSTM, we found these two methods have good prediction effects on the five different time-series data provided by the experiment. By comparing their experimental results, it is easy to find that the performance of the BARDR is better than the LSTM method. Except that the LSTM is a little better than the BARDR in the SP500 time-series prediction effect, the LSTM is worse than the BARDR in the other four time-series prediction effects. Although the BARDR is better than the LSTM in predicting performance, the BARDR method is several times longer than the LSTM method in experimental time. In addition, although in our experiments, the prediction values of the LSTM is worse than the BARDR method, the experimental time is short and the training parameters used are still very few. In future work, we may be able to improve performance of our proposed methods by increasing the number of neurons and the number of training iterations of the LSTM method. Based on the principle analysis, we think the predictive effect of the LSTM will exceed that of the BARDR method by reasonably increasing the number of neurons and the number of training iterations.

7. Experiments

We selected two types of experiment data to test in this paper in order to better evaluate the prediction effect of this method. The first type of experiment data is artificially simulated data generated automatically by computer. The

Table 4: The value of eight parameters of KNR.

Parameter name	Parameter type	Parameter value
*n*_neighbors	Int	5
Weights	"Uniform," or "distance"	Uniform
Algorithm	{"Auto," "ball_tree," "kd_tree," "brute"}	Auto
leaf_size	Int	30
p	Int	2
Metric	String or callable	Murkowski
metric_params	Dict	None
*n*_jobs	Int	1

Table 5: The length of the experiment data.

Name of data	Sin	SP500	HSI	DAX	ASX
All data length	10000	23204	8237	1401	4937
Train data length	9000	20884	7413	1261	4443
Test data length	1000	2320	824	140	494

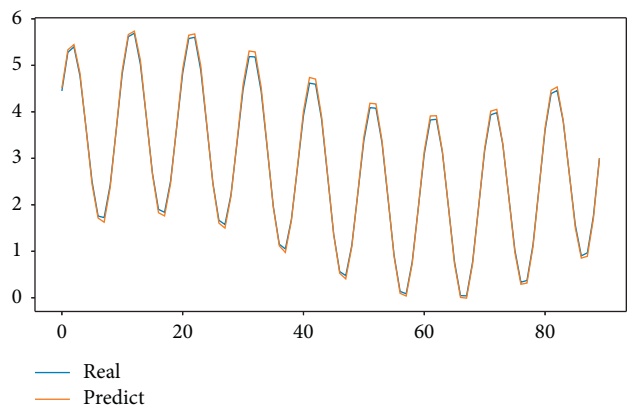


Figure 9: Prediction results of the LSTM method for sin sequence.

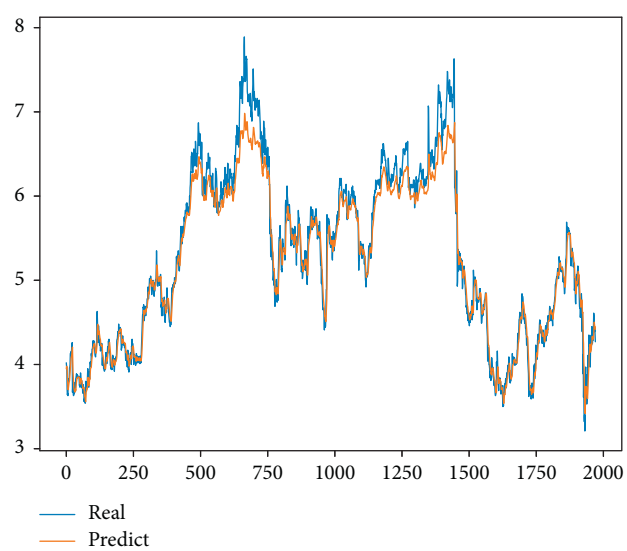


Figure 10: Prediction results of SVR for ASX sequence.

second type of experimental data is real-world stock index data. The model tested through social actual data is the most fundamental requirement for applying our proposed

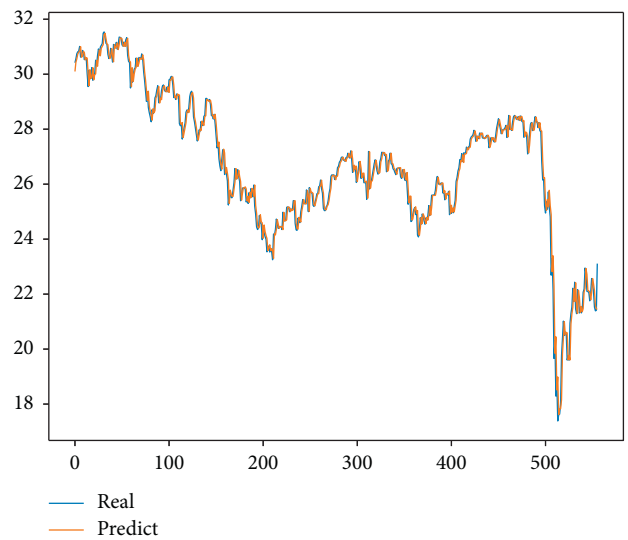


Figure 11: Prediction results of BARDR for DAX sequence.

method to some real-world fields. This section conducts detailed research and analysis on the experiment results of two aspects.

7.1. Analysis of Experimental Results Based on Artificial Simulation Data. We use artificially simulated experiment data to verify the effectiveness and correctness of our method. To get effective and accurate experiment result, the artificially generated simulation experiment data should be long enough. Hence, in the experiment, we choose the length of the artificially simulated experiment data to be 10,000, as shown in Table 5.

Before analyzing the experiment results of real-world stock index data, we first use the proposed LSTM-EMD and LSTM-EEMD methods to predict sin simulation sequence. The simulation experiment can verify the effectiveness and correctness of the two proposed methods.

Observing the sin data column of Table 1, we found the three indicators of the LSTM method, LSTM-EMD method, and LSTM-EEMD method are basically equivalent under the same number of neurons and iterations. The three results indicated that these three methods predict effects are similar to the sin simulation time-series data. Among them, the LSTM-EMD method has the best prediction effect, indicating that the method we proposed is effective and can improve the prediction effect of the experiment. Observing

the sin data column in Table 1, we found the experimental results of the LSTM are a little better than the LSTM-EEMD, but the gap is very small and can be almost ignored. After in-depth analysis, this is actually that the sin time-series data itself is very regular. By adding noise data and then using EEMD decomposition, the number of subtime series decomposition becomes more and more complicated, so the prediction effect is slightly worse. In summary, the two proposed prediction methods LSTM-EMD and LSTM-EEMD in this paper have a better comprehensive effect than the original LSTM method, indicating that the two proposed prediction methods are correct and effective.

7.2. Analysis of Experiment Results Based on Real Data. In order to verify the practicability of the proposed LSTM-EMD and LSTM-EEMD prediction methods, we use the two methods proposed in this paper to predict four real-world stock index time series in the financial market. The four time-series are the DAX, ASX, SP500, and HSI. The stock index time series is generated in human society and reflects social phenomena. The time series of stock indexes in the financial market is seriously affected by human factors, so it is complex and unstable. These four time-series are very representative. They come from four different countries or regions and can better reflect the changes in different countries in the world financial market.

By comparing the three evaluation indicators of RMSE, MAE, and R^2 of the LSTM, LSTM-EMD, and LSTM-EEMD prediction methods in Table 1, we found the prediction results of the LSTM-EEMD prediction method in the four sequence data are better than the LSTM-EMD prediction method. The experimental results show that the LSTM-EEMD prediction method is better than the LSTM-EMD prediction method. Figure 12 shows the prediction results of the LSTM-EMD method and the LSTM-EEMD method of HSI time series.

By observing Table 1, it is easy to find that the results of the proposed LSTM-EMD and LSTM-EEMD methods in the three sequences of HSI, DAX, and ASX are much better than the traditional LSTM method. We think that there are two main reasons for obtaining such good experimental results. On the one hand, the method proposed in this paper decomposes HSI, DAX, and ASX time series by EMD or EEMD so that the complex original HSI, DAX, and ASX time series are decomposed into multiple more regular and stable subtime series. The multiple subtime series are easier to predict than the complex original time series. Therefore, the predicting results of multiple subtime series is more accurate than the predicting results of directly predicting complex original time series. On the other hand, although the changes of HSI, DAX, and ASX time series are complicated, they can be decomposed into more regular and stable time series. In order to clearly understand the changes in HSI, DAX, and ASX time series and EMD or EEMD decomposition, Figures 13 and 14 show all changes in EMD or EEMD decomposition of DAX time series, respectively. Figure 8 is the original change of the DAX time series, and all subgraphs in Figures 13 and 14 are the EMD or EEMD decomposition

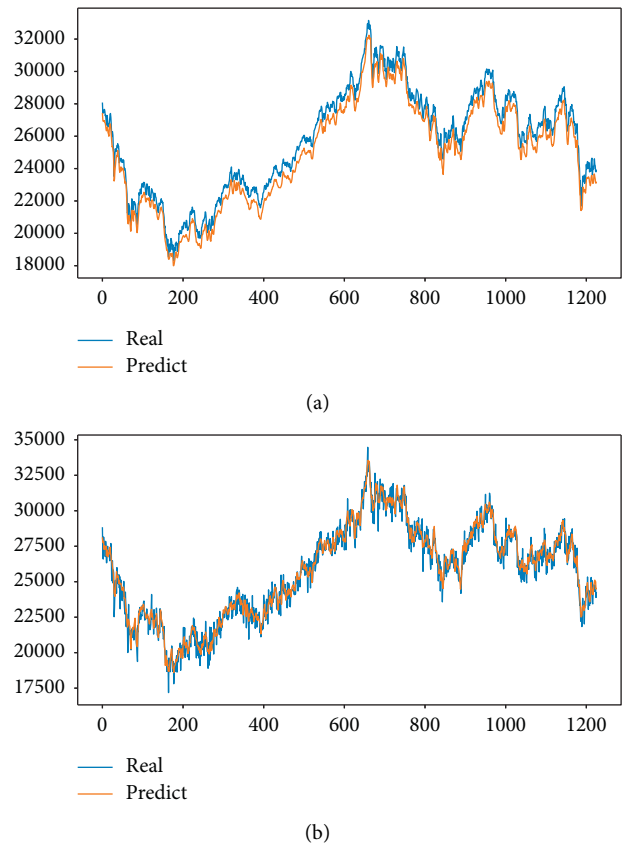


Figure 12: Prediction results of (a) LSTM-EMD method and (b) LSTM-EEMD method of HSI time series.

subgraphs of the DAX time series. It is easy to see that the lower EMD or EEMD decomposition subgraphs are gentler, more regular, more stable, more predictable, and have better prediction effects.

Carefully observe the data of the LSTM method, LSTM-EMD method, and LSTM-EEMD method in Table 1 in the three sequences of HSI, DAX, and ASX. It can be found that the proposed LSTM-EEMD method has a better experimental effect than the proposed LSTM-EMD method, and the proposed LSTM-EMD method has a better experimental effect than the traditional LSTM method. The three indicators RMSE, MAE, and R^2 used in the experiment all exemplify the above relationship. The smaller the RMSE or MAE experimental value of a method, the better the prediction effect of the method. However, the larger the R^2 value of a method, the better its prediction effect.

In this paper, we proposed a hybrid prediction method based on EMD or EEMD. The results of experiment show that the fusion prediction results are more superior to the traditional method for direct prediction of complex original time series in most instances. Although the experiments in this paper comparatively studied two hybrid methods based on EMD or EEMD and four other classical methods, such as SVR, BARDR, KNR, and LSTM. The two hybrid methods based on EMD and EEMD have different effects in different time series, and there are different time-series methods that reflect different experimental

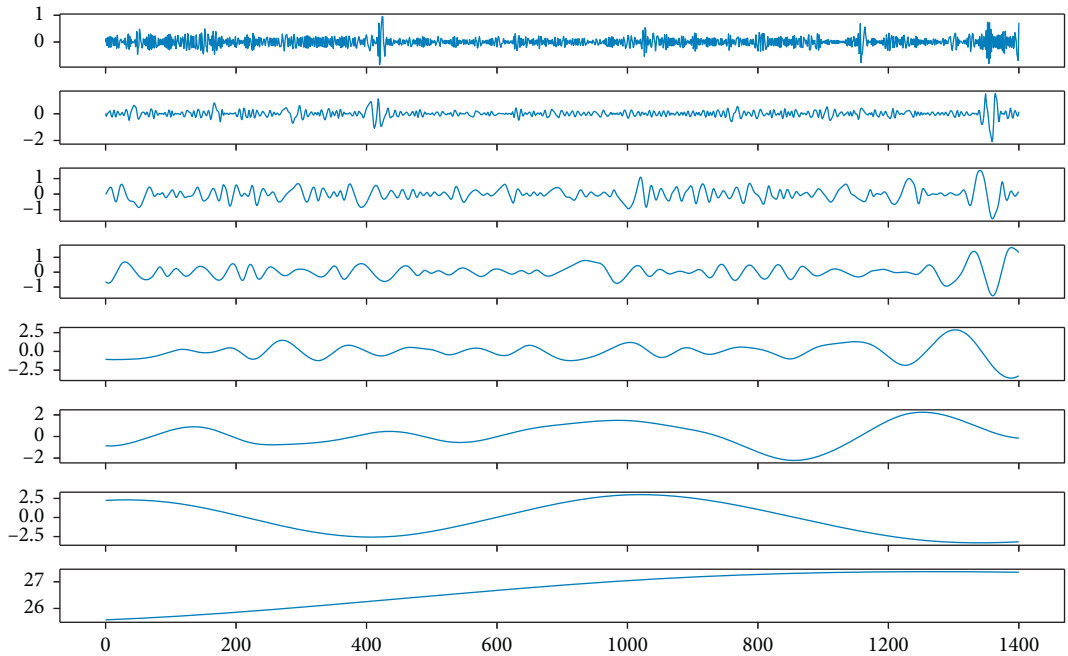


Figure 13: The EMD decomposition graphs of DAX time series.

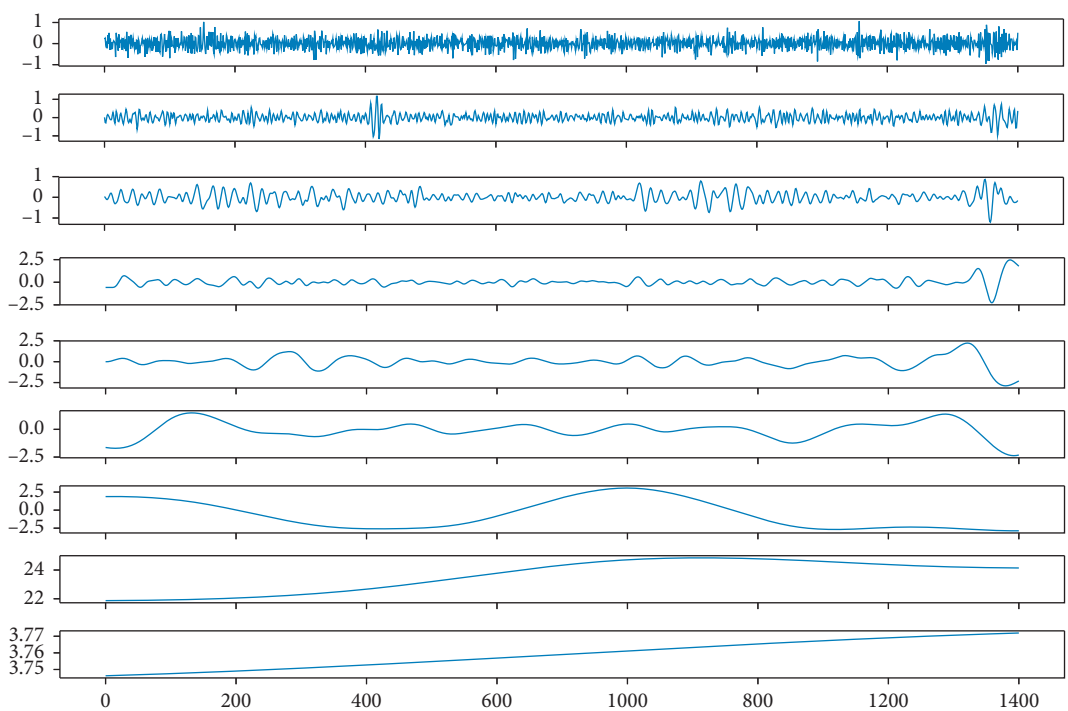


Figure 14: The EEMD decomposition graphs of DAX time series.

advantages and disadvantages. In actual application, people can choose different methods according to the actual situation and apply it to specific practical fields. In the actual environment, if the results obtained by the method you choose do not meet your expectations, you can choose another method.

8. Conclusion and Future Work

We proposed a hybrid short-term prediction method LSTM-EMD or LSTM-EEMD based on LSTM and EMD or EEMD decomposition methods. The method is based on a complex problem divided and conquered strategy. Combining the

advantages of EMD, EEMD, and LSTM, we used the EMD or EEMD method to decompose the complex sequence into multiple relatively stable and gentle subsequences. Use the LSTM neural network to train and predict each subtime series. The prediction process is simple and requires only two steps to complete. First, we use the LSTM to predict each subtime series value. Then, the prediction results of multiple subtime series are fused to form a complex original time-series prediction result. In the experiment, we selected five data for testing to fully the performance of the method. The comparison results with the other four prediction methods show that the predicted values show higher accuracy. The hybrid prediction method we proposed is effective and accurate in future stock price prediction. Hence, the hybrid prediction method has practical application and reference value. However, there are some shortcomings. The proposed method has some unexpected effects on the experimental results of time series with very orderly changes.

The research and application of analysis and prediction methods on time series have a long history and rapid development, but the prediction effect of traditional methods fails to meet certain requirement of real application on some aspects in certain fields. Improving the prediction effect is the most direct research goal. Taking the study results of this paper as a starting point, we still have a lot of work in the future that needs further research. In the future, we will combine the EMD method or EEMD method with other methods, or use the LSTM method in combination with wavelet or VMD.

Data Availability

Data are fully available without restriction. The original experimental data can be downloaded from Yahoo Finance for free (http://finance.yahoo.com).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Yang Yujun contributed to all aspects of this work. Yang Yimei and Xiao Jianhua conducted the experiment and analyzed the data. All authors reviewed the manuscript.

Acknowledgments

This work was supported in part by the Scientific Research Fund of Hunan Provincial Education under Grants 17C1266 and 19C1472, Key Scientific Research Projects of Huaihua University under Grant HHUY2019-08, Key Laboratory of Wuling-Mountain Health Big Data Intelligent Processing and Application in Hunan Province Universities, and the Key Laboratory of Intelligent Control Technology for Wuling-Mountain Ecological Agriculture in Hunan Province under Grant ZNKZ2018-5. The fund source has no role in research design, data collection, analysis or interpretation, or the writing of this manuscript.

References

[1] E. Hadavandi, H. Shavandi, and A. Ghanbari, "A genetic fuzzy expert system for stock price forecasting," in *Proceedings of the 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery*, pp. 41–44, Yantai, China, August 2010.

[2] C. Zheng and J. Zhu, "Research on stock price forecast based on gray relational analysis and ARMAX model," in *Proceedings of the 2017 International Conference on Grey Systems and Intelligent Services (GSIS)*, pp. 145–148, Stockholm, Sweden, August 2017.

[3] A. Kulaglic and B. B. Üstündağ, "Stock price forecast using wavelet transformations in multiple time windows and neural networks," in *Proceedings of the 2018 3rd International Conference on Computer Science and Engineering (UBMK)*, pp. 518–521, Sarajevo, Bosnia, September 2018.

[4] G. Xi, "A novel stock price forecasting method using the dynamic neural network," in *Proceedings of the 2018 International Conference on Robots & Intelligent System (ICRIS)*, pp. 242–245, Changsha, China, February 2018.

[5] Y. Yu, S. Wang, and L. Zhang, "Stock price forecasting based on BP neural network model of network public opinion," in *Proceedings of the 2017 2nd International Conference on Image, Vision and Computing (ICIVC)*, pp. 1058–1062, Chengdu, China, June 2017.

[6] J.-S. Chou and T.-K. Nguyen, "Forward forecast of stock price using sliding-window metaheuristic-optimized machine-learning regression," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 3132–3142, 2018.

[7] Y. Guo, S. Han, C. Shen, Y. Li, X. Yin, and Y. Bai, "An adaptive SVR for high-frequency stock price forecasting," *IEEE Access*, vol. 6, pp. 11397–11404, 2018.

[8] J. Lee, R. Kim, Y. Koh, and J. Kang, "Global stock market prediction based on stock chart images using deep Q-network," *IEEE Access*, vol. 7, pp. 167260–167277, 2019.

[9] J. Zhang, Y.-H. Shao, L.-W. Huang et al., "Can the exchange rate be used to predict the shanghai composite index?" *IEEE Access*, vol. 8, pp. 2188–2199, 2020.

[10] Y. Guo, S. Han, C. Shen, Y. Li, X. Yin, and Y. Bai, "An adaptive SVR for high-frequency stock price forecasting," *IEEE Access*, vol. 6, pp. 11397–11404, 2018.

[11] D. Lien Minh, A. Sadeghi-Niaraki, H. D. Huy, K. Min, and H. Moon, "Deep learning approach for short-term stock trends prediction based on two-stream gated recurrent unit network," *IEEE Access*, vol. 6, pp. 55392–55404, 2018.

[12] I. I. Hassan, "Exploiting noisy data normalization for stock market prediction," *Journal of Engineering and Applied Sciences*, vol. 12, no. 1, pp. 69–77, 2017.

[13] S. Jeon, B. Hong, and V. Chang, "Pattern graph tracking-based stock price prediction using big data," *Future Generation Computer Systems*, vol. 80, pp. 171–187, 2018.

[14] E. Chong, C. Han, and F. C. Park, "Deep learning networks for stock market analysis and prediction: methodology, data representations, and case studies," *Expert Systems with Applications*, vol. 83, pp. 187–205, 2017.

[15] X. Ding, Y. Zhang, T. Liu, and J. Duan, "Using structured events to predict stock price movement: an empirical investigation," in *Proceedings of the EMNLP*, pp. 1–11, Doha, Qatar, October 2014.

[16] M. Dang and D. Duong, "Improvement methods for stock market prediction using financial news articles," in *Proceedings of 2016 3rd National Foundation for Science and Technology Development Conference on Information and*

Computer Science, pp. 125–129, Danang City, Vietnam, September 2016.

[17] B. Weng, M. A. Ahmed, and F. M. Megahed, "Stock market one-day ahead movement prediction using disparate data sources," *Expert Systems with Applications*, vol. 79, pp. 153–163, 2017.

[18] Y. Yujun, L. Jianping, and Y. Yimei, "An efficient stock recommendation model based on big order net inflow," *Mathematical Problems in Engineering*, vol. 2016, Article ID 5725143, 15 pages, 2016.

[19] S. O. Ojo, P. A. Owolawi, M. Mphahlele, and J. A. Adisa, "Stock market behaviour prediction using stacked LSTM networks," in *Proceedings of the 2019 International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*, pp. 1–5, Vanderbijlpark, South Africa, November 2019.

[20] S. Liu, G. Liao, and Y. Ding, "Stock transaction prediction modeling and analysis based on LSTM," in *Proceedings of the 2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pp. 2787–2790, Wuhan, China, May 2018.

[21] D. Wei, "Prediction of stock price based on LSTM neural network," in *Proceedings of the 2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM)*, pp. 544–547, Dublin, Ireland, October 2019.

[22] K. Chen, Y. Zhou, and F. Dai, "A LSTM-based method for stock returns prediction: a case study of China stock market," in *Proceedings of the 2015 IEEE International Conference on Big Data (Big Data)*, pp. 2823-2824, Santa Clara, CA, USA, November 2015.

[23] Y. Yang and Y. Yang, "Hybrid method for short-term time series forecasting based on EEMD," *IEEE Access*, vol. 8, pp. 61915–61928, 2020.

[24] A. H. Bukhari, M. A. Z. Raja, M. Sulaiman, S. Islam, M. Shoaib, and P. Kumam, "Fractional neuro-sequential ARFIMA-LSTM for financial market forecasting," *IEEE Access*, vol. 8, p. 71326, 2020.

[25] S. Ma, L. Gao, X. Liu, and J. Lin, "Deep learning for track quality evaluation of high-speed railway based on vehicle-body vibration prediction," *IEEE Access*, vol. 7, pp. 185099–185107, 2019.

[26] Y. Hu, X. Sun, X. Nie, Y. Li, and L. Liu, "An enhanced LSTM for trend following of time series," *IEEE Access*, vol. 7, pp. 34020–34030, 2019.

[27] S. Hochreiter and J. Schmidhuber, "Long short-term mfemory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[28] Y. Yang, J. Li, and Y. Yang, "The cross-correlation analysis of multi property of stock markets based on MM-DFA," *Physica A: Statistical Mechanics and Its Applications*, vol. 481, pp. 23–33, 2017.

[29] Y. Yujun, L. Jianping, and Y. Yimei, "Multiscale multifractal multiproperty analysis of financial time series based on Rényi entropy," *International Journal of Modern Physics C*, vol. 28, no. 2, 2017.

[30] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000*, vol. 3, pp. 189–194, Como, Italy, July 2000.

[31] Y. Wei, Z. Wang, M. Xu, and S. Qiao, "An LSTM method for predicting CU splitting in H. 264 to HEVC transcoding," in *Proceedings of the IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–4, St. Petersburg, FL, USA, December 2017.

[32] C. Zhang, W. Ren, T. Mu, L. Fu, and C. Jia, "Empirical mode decomposition based background removal and de-noising in polarization interference imaging spectrometer," *Optics Express*, vol. 21, no. 3, pp. 2592–2605, 2013.

[33] X. Zhou, H. Zhao, and T. Jiang, "Adaptive analysis of optical fringe patterns using ensemble empirical mode decomposition algorithm," *Optics Letters*, vol. 34, no. 13, pp. 2033–2035, 2009.

[34] N. E. Huang, J. R. Yeh, and J. S. Shieh, "Ensemble empirical mode decomposition: a noise-assisted data analysis method," *Advances in Adaptive Data Analysis*, vol. 1, no. 1, pp. 1–41, 2009.

[35] H. Zhang, F. Wang, D. Jia, T. Liu, and Y. Zhang, "Automatic interference term retrieval from spectral domain low-coherence interferometry using the EEMD-EMD-based method," *IEEE Photonics Journal*, vol. 8, no. 3, pp. 1–9, Article ID 6900709, 2016.

[36] Y. Jiang, C. Tang, X. Zhang, W. Jiao, G. Li, and T. Huang, "A novel rolling bearing defect detection method based on bispectrum analysis and cloud model-improved EEMD," *IEEE Access*, vol. 8, pp. 24323–24333, 2020.

[37] W. Wang, W. Peng, M. Tian, and W. Tao, "Partial discharge of white noise suppression method based on EEMD and higher order statistics," *The Journal of Engineering*, vol. 2017, no. 13, pp. 2043–2047, 2017.

[38] Z. Wei, K. G. Robbersmyr, and H. R. Karimi, "An EEMD aided comparison of time histories and its application in vehicle safety," *IEEE Access*, vol. 5, pp. 519–528, 2017.

[39] Y. Yang and Y. Yang, "Hybrid prediction method for wind speed combining ensemble empirical mode decomposition and Bayesian ridge regression," *IEEE Access*, vol. 8, pp. 71206–71218, 2020.

[40] J. Sun and H. Sheng, "Applications of Ensemble Empirical mode decomposition to stock-futures basis analysis," in *Proceedings of the 2010 2nd IEEE International Conference on Information and Financial Engineering*, pp. 396–399, Chongqing, China, September 2010.

[41] B. Al-Hnaity and M. Abbod, "A novel hybrid ensemble model to predict FTSE100 index by combining neural network and EEMD," in *Proceedings of the 2015 European Control Conference (ECC)*, pp. 3021–3028, Linz, Austria, July 2015.

[42] N. E. Huang, Z. Shen, S. R. Long et al., "The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis," *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 454, no. 1971, pp. 903–995, 1998.

[43] F. Jiang, Z. Zhu, and W. Li, "An improved VMD with empirical mode decomposition and its application in incipient fault detection of rolling bearing," *IEEE Access*, vol. 6, pp. 44483–44493, 2018.

[44] Z. H. Wu and N. E. Huang, "Ensemble empirical mode decomposition: a noise-assisted data analysis method," *Advances in Adaptive Data Analysis*, vol. 1, no. 1, pp. 1–14, 2009.

Hindawi
Complexity
Volume 2020, Article ID 8855885, 10 pages
https://doi.org/10.1155/2020/8855885

Research Article

Feature Guided CNN for Baby's Facial Expression Recognition

Qing Lin,[1] **Ruili He**,[2] **and Peihe Jiang**[3]

[1]*Integrated Information Center of Yantai, Yantai 264003, China*
[2]*School of Computer Science and Technology, Harbin Institute of Technology, Weihai 264209, China*
[3]*School of Opto-Electronic Information Science and Technology, Yantai University, Yantai 264005, China*

Correspondence should be addressed to Peihe Jiang; jiangpeihe@163.com

Received 5 September 2020; Revised 23 October 2020; Accepted 5 November 2020; Published 23 November 2020

Academic Editor: Min Xia

State-of-the-art facial expression methods outperform human beings, especially, thanks to the success of convolutional neural networks (CNNs). However, most of the existing works focus mainly on analyzing an adult's face and ignore the important problems: how can we recognize facial expression from a baby's face image and how difficult is it? In this paper, we first introduce a new face image database, named BabyExp, which contains 12,000 images from babies younger than two years old, and each image is with one of three facial expressions (i.e., happy, sad, and normal). To the best of our knowledge, the proposed dataset is the first baby face dataset for analyzing a baby's face image, which is complementary to the existing adult face datasets and can shed some light on exploring baby face analysis. We also propose a feature guided CNN method with a new loss function, called distance loss, to optimize interclass distance. In order to facilitate further research, we provide the benchmark of expression recognition on the BabyExp dataset. Experimental results show that the proposed network achieves the recognition accuracy of 87.90% on BabyExp.

1. Introduction

Facial expressions play an important role in human being's communication. The ability to differentiate genuine displays of emotional experience from the posed ones is very important for dealing with day-to-day social interactions. Humans and computer algorithms can greatly benefit from being able to distinguish the genuine expression from the posed one. Possible applications of automated facial expression recognition include better transcription of videos, movies, or advertisement recommendations and detection of pain in telemedicine. Therefore, facial expression recognition has attracted a vast amount of attention in the past two decades [1–6]. The development of facial expression recognition relies heavily on an adequate database of facial expressions. However, due to the nature of facial expressions, there are a limited number of publicly available databases providing a sufficient number of facial images tagged with accurate expression information. Table 1 shows the major differences of the existing image databases with the number of images, number of subjects, expression distribution, data size, and the released years. However, most of the existing works and datasets [7–11] focus on analyzing adult faces, which ignore how to analyze facial expressions from baby facial images. Although some datasets include children, there are actually very few images of very young children. None of these datasets is specifically designed to explore the expression of babies. There are two main reasons for the lack of research on baby face analysis. The first reason is that the community has not realized the application values of analyzing baby's facial expression. In fact, there are many applications of analyzing the facial expressions of babies, such as advertising marketing for parents, intelligent family child care, and scientific parenting. The second reason may be traced to the additional challenge of obtaining the baby face datasets with accurate expression labels.

As we all know, 0–2 years old is a golden period for the development of a baby and for laying a solid foundation for their lifelong physical and mental health. Therefore, it is valuable to develop the algorithm to interpret a baby's facial expressive signals for scientific parenting. In addition, due to the support of national policies and people's growing

Table 1: Overview of the existing facial expression datasets.

Datasets	JAFFE	PIE	MMI	BU-3DFE	CK+	FER	CAFE	SFEW	RAF-DB
Images	213	40,000	740	2500	593	35,887	1192	1766	29,672
Subjects	10	68	25	100	137	—	100	—	—
Class	7	4	—	7	7	7	7	7	7
Size	256 x 256	—	720 x 576	—	640 x 480	48 x 48	Square	720 x 576	—
Age	—	—	19–62	18–70	18–50	—	2–8	—	0–70
Gender	Female	Both	Both	Both	Both	Both	Both	Both	Both
Year	1988	2000	2005	2006	2010	2013	2014	2015	2017

attention to the growth and development of a baby, the parenting market has been expanding. Accurate recognition of facial expressions of a baby is of great significance to facilitate the development of scientific parenting. All these real needs have brought a strong motivation to the study of recognizing baby's face expressions.

Recently, researchers have realized the importance of children's facial expressions in order to study developmentally the interpretation of these expression datasets. For example, the new NIMH Children's Emotional Face Picture Collection (NIMH-ChEFS) contains photos of children aged 10–17 [12], the Radboud Faces Database includes photos of 8- to 12-year-olds [13], and the CAFE set features photographs of 2- to 8-year-old children [14]. Although these new datasets give researchers the option to use a sample of children aged 2–17 years, there have been no datasets that feature smaller children to date. On the contrary, all the datasets mentioned above for children's facial expressions have only a small number of images, which are not suitable for training convolutional neural network (CNN) models. In addition, these datasets contain the facial images with posed expressions in a lab-controlled environment.

In this paper, to address the aforementioned issues, we propose a new image dataset with expression labels of baby faces for automatic facial expression recognition. Our dataset, which is called the BabyExp dataset, contains more than 12,000 images from babies younger than two years old showing spontaneous expressions in an uncontrolled environment. Each face image is annotated with one of three facial expressions (i.e., happy, sad, and normal). It is complementary to existing adult face datasets and can shed some light on exploring baby face analysis. Our key contributions are summarized as follows:

(1) We present a facial expression dataset, named BabyExp, which contains more than 12,000 images from babies showing spontaneous genuine expressions in an uncontrolled environment. Each image is annotated with one of three facial expressions (i.e., happy, sad, and normal).

(2) We propose a new distance loss function to effectively enhance the discriminative ability of distance between classes in unconstrained facial expression recognition tasks.

(3) In order to facilitate further research, we proposed a new method for facial analysis and evaluated its performance on the BabyExp dataset. Experimental results show that the proposed network achieves a recognition accuracy of 87.90% on the test set of BabyExp.

2. Materials and Methods

2.1. Data Collection. Our baby face images are generated from both static images and video sequences uploaded by parents using smartphones. We will introduce the preprocessing of the BabyExp dataset in the following. For the original images and the original video data, we first perform face detection, then perform face cropping, and finally perform picture similarity detection. A detailed description can be found in the following.

2.1.1. Image Preprocessing. For image processing, we first use the Dlib visual library [15] and the OpenCV visual library to perform face detection and cropping on the original image. During the face detection, we adopt the following strategy. First of all, if a face appears, the face section will be extracted. Second, if no face is detected during the detection, we rotate the image 270 degrees clockwise at 90 degrees each time. If a face appears during the three rotation detection processes, then we crop and save the face image. Last, if there are two or more faces detected in the image, we will assume that this image will have an adult face or a face that is not a human face but is misidentified as a human face. Then, we will discard such images.

It is important to note that the area of the original picture of the baby's face is not very large. At this point, the picture is redundant. If it is used directly for training, the model converges slowly, resulting in poor test results. In order to reduce the large amount of nonface information in the image, therefore, after using the above Dlib face detection strategy, when cropping the face, we crop the face area according to a specific artificial strategy and save it. The main purpose is to obtain a noise-free and good-quality baby face image dataset in order to obtain a better model during the training process and a better accuracy during the test process. We then crop the original image according to the new picture size and finally normalize the cropped image (the normalized size is 256×256).

2.1.2. Video Preprocessing. We segment the original video data, take an image every 30 frames, and then perform the same process as the static image data preprocessing on the images from the video frames, detecting, rotating, and finally cropping the baby's face picture. It should be noted that

because the pictures obtained by intercepting video frames may have great similarities, many images are redundant, so the only different operation different from the static image is that, after the picture is cropped and saved, we need to perform picture similarity matching operations to filter the image. We use SSIM [16] to perform similarity matching and specify to delete images with similarity greater than 90%.

2.2. Data Annotation. After preprocessing, we get 7,600 images, and we will tag the images with facial expressions. Because babies are all at the stage of 0–2 years old, their expressions are not as diverse as those of the adults. For this reason, we specially selected three main baby expressions (i.e., normal, sad, and happy) for the BabyExp dataset. The marking process is divided into three steps: manual labeling, label statistical analysis, and label aggregation.

In the manual labeling step, 10 raters coming from Harbin Institute of Technology were selected to manually label the data. Without given any information, the subjects were asked to classify the photos according to their own experience. In order to save time and to boost classification efficiency, we used C++ language to design a manual labeling tool for manual classification and record the human evaluator choice of the expression label. For each input image, we asked 10 raters to label the image into one of 3 emotion types and 1 error fold: happy, sad, normal, and error. The raters are required to choose one single emotion for each image. After labeling, there will be four categories, i.e., happy, normal, sad, and error. The error category represents that an image is not a human face or the face is unclear.

The second step is to label statistical analysis. After the manual labeling of 10 people is completed, it is necessary to analyze the expressions in all the categories. The statistical result is an expression category selected by 10 people per picture. With labels from 10 raters for each face image, we can generate a probability distribution of emotion captured by the facial expression. Let N denote the number of the training examples $I_i, i = 1, \ldots, N$. Given the i-th example I_i, its label distribution from the raters can be expressed as $p_k^i, k = 1, \ldots, 4$. Naturally, we have

$$\sum_{k=1}^{4} p_k^i = 1. \tag{1}$$

The final step is to aggregate the labels of each image. After the second step, we need to aggregate the label of each expression generated by the 10 people. The combined labeling results are happy, normal, sad, and error. In most of the existing facial expression datasets, each facial image is only associated with one single label. If the image has more than one label, it is natural to assign the image to the label of the largest p_k^i. We experimented majority voting schemes. More formally, we create a new target distribution.

$$\widehat{p}_k^i = \begin{matrix} 1, & \text{if } k = \operatorname{argmax}_j p_j^i, \\ 0, & \text{otherwise.} \end{matrix} \tag{2}$$

After processing, when encountering an image, a certain type of expression will be selected, which means that the image is the corresponding category. If an image has the same labeling number of people and both have the maximum number of votes, the image is not classified, and they are marked twice to determine the baby's expression label of the image. Finally, in the end, we obtained 2,502 happy images, 4,028 normal images, and 1,070 sad images, as shown in Figure 1. It can be clearly seen that the three expression distributions in the baby expression dataset are unbalanced. This is because babies are different from adults who have rich expressions leading to a uniform expression distribution. Since expressions of babies from 0 to 2 years old are still developing and the expression types are relatively monotonous, especially in the absence of outside interference, most of the time, the baby is in a calm state followed by the state of laughter and finally, the state of sadness, so we can see that the proportion of normal is relatively large, and the proportion of sad is relatively small, which is very consistent with the expression characteristics of the baby, but imbalanced data may have a strong impact on the accuracy of the research experiment results; one solution is to use data augmentation and synthesis to balance the distribution of classes during the preprocessing phase.

2.3. Data Augmentation. According to the dataset information obtained above, there is an imbalance in the dataset, which will adversely affect the subsequent experimental work. Although deep learning has a strong characteristic learning ability, some technical hurdles prevent their successful applications to our dataset. First, deep neural networks require a lot of training data to avoid overfitting. Additionally, models trained using imbalance facial expression samples have a poor generalization ability and are prone to overfitting, which is illustrated in the experiments we introduced later in the experimental section. So, we need to perform data augmentation to promote data balance and facilitate the use of deep learning methods for experiments.

At present, generative adversarial networks (GANs) [17] are a popular research method in the field of machine learning. Their basic idea is derived from the game of two players in game theory. In the GAN framework, a "generator" network is tasked with fooling a "discriminator" network into believing that its own samples are real data. Inspired by the successful application of the GAN in the field of image style transfer, this project will use the GAN as a network model for image enhancement processing. We can use the resulting generative model to generate faces with specific expressions from nothing but random noise. Many different types of GANs require paired datasets for image style transfer. Baby expression images do not have paired data for sad and happy expressions corresponding to the same normal expressions of the baby, so the research contents in this part will draw on the important idea of CycleGAN [18] asymmetry training for unpaired image-to-image translation. The research contents in this part mainly include data augmentation of sad and happy facial expression images for imbalanced baby facial expression data based on CycleGAN.

The CycleGAN architecture contains two generators and two adversarial discriminators: Generator A, Generator B,

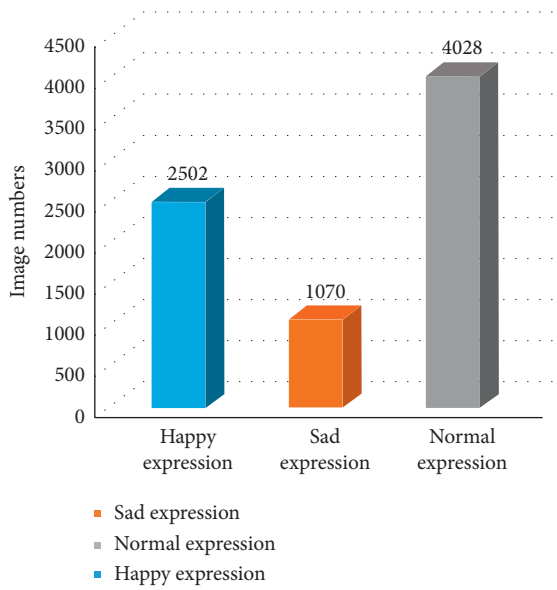


FIGURE 1: Image numbers of three expressions after preprocessing.

Discriminator A, and Discriminator B, where Generator A tries to generate images Generated_B that look similar to images from domain B, while Discriminator B aims to distinguish between translated samples Generated_B and real samples B. The overall structure of the algorithm in our data augmentation design is shown in Figure 2. Generator A inputs normal expression image A and output happy expression image Generated_B. Cyclic_A generated by Generator B brings Generated_B back to the original normal expression image A, where Cyclic_A is called the cyclic image of A. Generator B inputs happy expression image B and outputs normal expression image Generated_A. Cyclic_B is generated by Generator A, and Generated_A is brought back to the original happy expression image B. Cyclic_B is called the circular image of B. Discriminator A is used to distinguish true or false of the input normal expression image, and Discriminator B is used to distinguish true or false of the input happy expression image, respectively. Similarly, the data augmentation of sad expressions has the same process structure as that of happy expressions, which is not described in detail here.

It must be pointed out that because the number of normal expressions is sufficient, we have only enhanced the sad and happy expression image data. Finally, after data augmentation of CycleGAN, 1,498 happy expression images and 2,955 sad expression images are finally selected and generated. The total amount of facial expression data we obtained is shown in Table 2. It can be seen that, after data augmentation, we obtained 4,000 happy images, 4,028 normal images, and 4,025 sad images. We have a total of 12,053 baby facial expression images. We call it the BabyExp dataset, of which 4,453 are generated images. The amount of data for three facial expressions has reached an equilibrium state for the future academic research.

2.4. Proposed Methods. The overall pipeline of the proposed deep learning approach is depicted in Figure 3. Our proposed framework, called VFESO-DLSE, is composed of four modules: feature extraction, feature refinement, covariance pooling, and CNN classification. We also propose a new loss function, called distance loss, denoted as $\mathscr{L}_{DL}$.

2.4.1. Distance Loss. Min Xia et al. [19] found that the feature constraint helps enlarge the feature distance of different age range feature space in face images with similar feature distributions. Inspired by this, we propose a novel loss function, called distance loss, which takes strong feature constraint into baby facial expression learning. The distance loss aims to learn representations with lower intraclass variations and higher interclass distances. As we all know, by pushing the samples to the corresponding class center in the feature space during the training, the center loss [20] significantly reduces the intraclass difference. The center loss is defined as the sum of the square distance between the sample and its corresponding class center in the feature space. The center loss is denoted as $\mathscr{L}_C$:

$$\mathscr{L}_C = \frac{1}{2}\sum_{i=1}^{m}\left\|x_i - c_{y_i}\right\|^2, \tag{3}$$

where y_i is the class label of the i-th sample; x_i denotes the feature vector of the i-th sample taken from the FC layer before the decision layer; c_{y_i} denotes the center of all the samples with the same class label as x_i; and m is the number of samples in the mini-batch. Our distance loss denoted as $\mathscr{L}_{DL}$ is defined as

$$L_{DL} = L_C + \lambda_1 \sum_{C_j \in N_j} \sum_{\substack{C_k \in N_k \\ C_k \neq C_j}} \left(\frac{1}{\left\|C_k - C_j\right\|_2 + 1}\right), \tag{4}$$

where N_j and N_k denote the set of expression labels and C_k and C_j denote the k-th and j-th centers. Specifically, the first term was used to narrow the distance between the sample and the center of the corresponding class, and the second term was used to punish the similarity between different expressions. λ_1 is used to balance the weights of the two terms. By minimizing the distance loss function, the same expression will be brought closer, and different expressions will be pushed in the feature space.

2.4.2. Feature Guided CNN. As we all know, the expression change of babies aged 0 to 2 years will be less distorted. Although CNNs have achieved great performance in image processing [21–23], traditional CNNs consist of fully connected layers, maximum or average poolings, and convolutional layers to capture only first-order information [24]. We believe that second-order statistics is more suitable to capture such baby's expression distortions than first-order statistics. So, we take network architecture model-4 presented in [25] as a baseline model. Related studies [26, 27] have proved that the trained deep convolutional network can be used as a feature extraction tool for classification tasks, and it has a generalization ability. Following up this idea, we apply the famous VGG16 [28] model for feature extraction

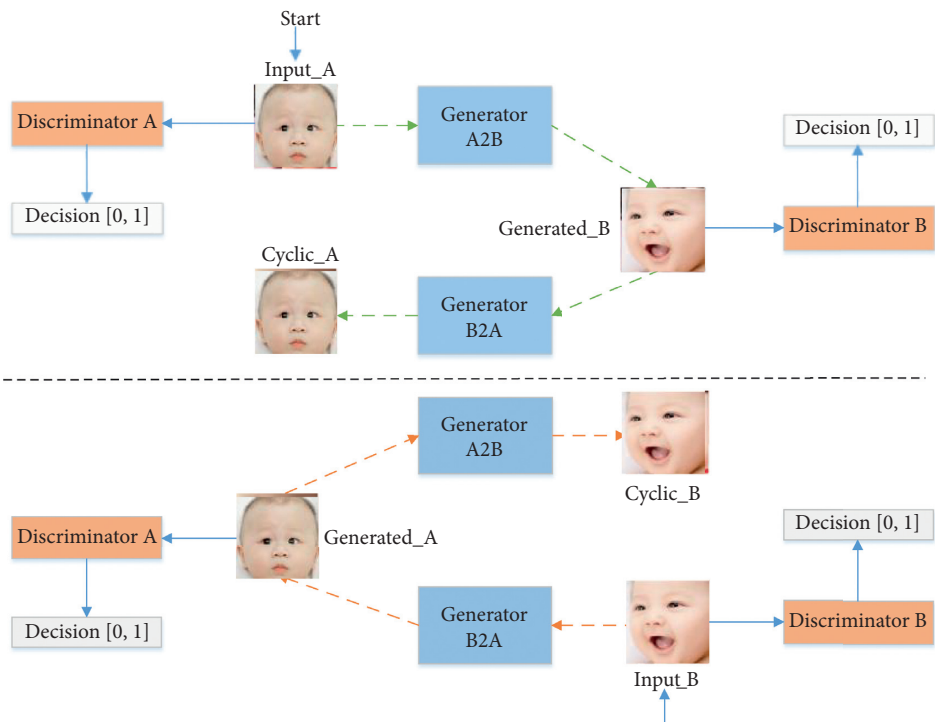


Figure 2: Example of happy expression image data augmentation process.

Table 2: Expression details for the BabyExp dataset.

Total images	Expression class	Age (month)	Happy images	Sad images	Normal images
12,053	3	0–24	4000	4025	4028

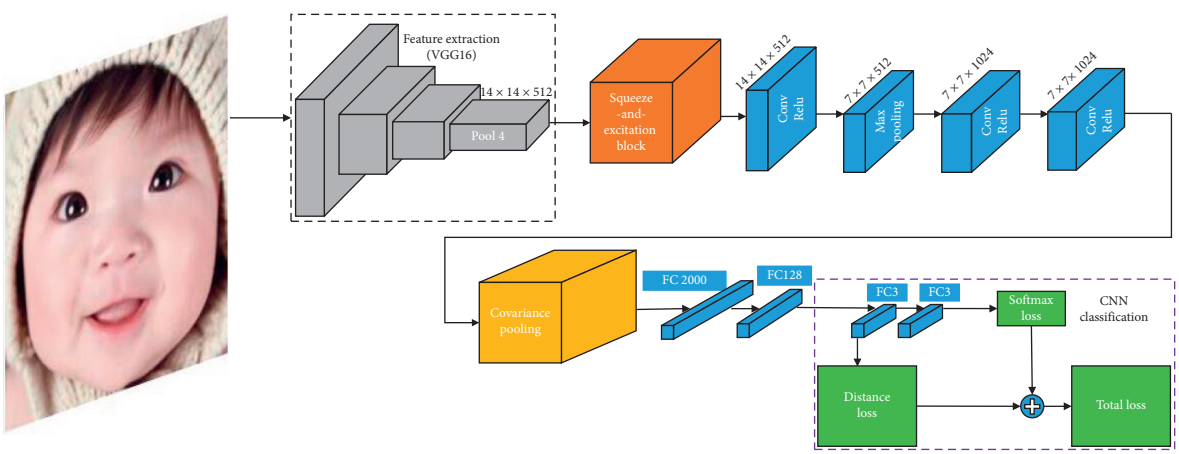


Figure 3: The overall architecture of the proposed deep learning approach VFESO-DLSE.

in our method. VGG16 is a typical CNN model. It has 13 convolutional layers, 5 pooling layers, and 3 fully connected layers for face recognition. To extract expression features, we use a pretrained VGG16 network on the expression dataset to extract features (referred to as VFE). For each facial image, we use the $14 \times 14 \times 512$ size feature maps of the fourth pooling layer to represent an image feature.

For the feature refinement stage, we use the squeeze-and-excitation (SE) block [29] to refine the CNN functionality and highlight the regions of expression that need to be highlighted, thereby explicitly modeling the interdependencies between the channels by adaptively recaliberating the channel's feature response. The detailed structure can be seen in Figure 4, and γ is a scaling parameter (16 in this paper). The purpose of this parameter is to reduce the number of channels and thus reduce the computation. C represents the number of channels, and H, W represent the height and width of the feature map

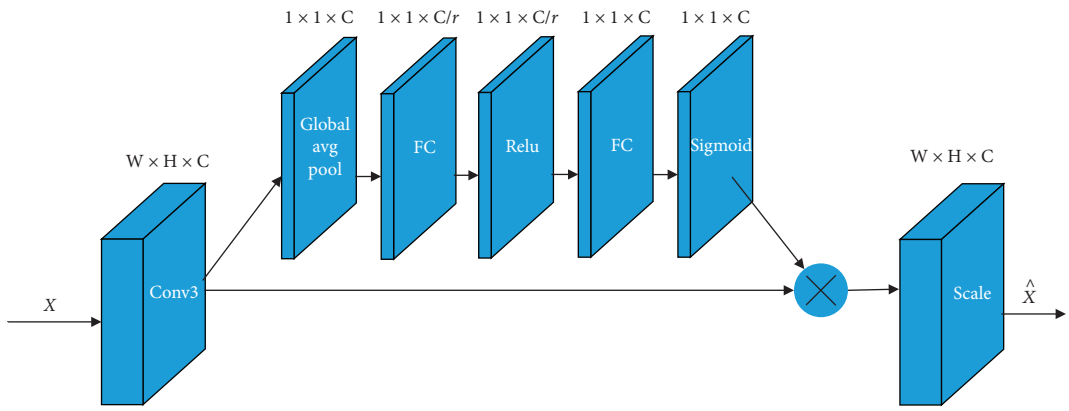


Figure 4: Squeeze-and-excitation (SE) block; γ is a scaling parameter.

input from the previous layer. The SE module first performs a squeeze operation on the feature map obtained by the convolution to obtain channel-level global features; here, we use global average pooling as the squeeze operation. Then, an excitation operation is performed on the global features. Two fully connected layers form a bottleneck structure, and the correlation between the channels is modeled. The number of output weights is the same as the number of input features. As shown in Figure 4, we first reduce the feature dimension to 1/16 of the input and then activate it through ReLu and then rise back to the original dimension through a fully connected layer, which learns the relationship between each channel and also obtains the weight of different channels and finally multiplies the original feature map to get the final feature. In essence, the SE module performs attention or gating operations on the channel dimension. This attention mechanism allows the model to pay more concern about the channel features with the most information and suppress those unimportant channel features.

Then, three convolutions with kernel size 3×3 are followed, and we use ReLU [20] as the activation function for each convolution layer and two max pooling layers. Then, the same as baseline [25], we also use covariance pooling after the last convolutional layer and before the fully connected layers. In the last classification part, the total loss of our network architecture training is formulated as follows:

$$\mathscr{L}_{\text{total}} = \mathscr{L}_s + \lambda \mathscr{L}_{DL}, \tag{5}$$

where $\mathscr{L}_s$ denotes the softmax loss and $\mathscr{L}_{DL}$ denotes the distance loss. The hyper parameter λ is used to balance the two loss functions.

2.5. Experiments

2.5.1. Experimental Setup. All the training and testing are carried out on the NVIDIA GeForce GTX 1080Ti 16G GPUs. We use deep learning framework TensorFlow [30] to develop the model. On an Ubuntu Linux system with NVIDIA GPUs, it takes 10–15 hours to train a model based on our network structures.

2.5.2. Implementation Details. We set up three major experiments: the first experiment is to evaluate the state-of-the-art adult facial expression analysis methods on BabyExp to see if the adult expression recognition method works for baby images. In this part, we use the methods trained on SFEW2.0 and test on BabyExp, and Table 3 shows the results of this experiment.

The second experiment is to demonstrate the effectiveness of the proposed method VFESO-DLSE. We compare our method against four designed architectures: DLP [31], the baseline [25], baseline + distance loss (SO-DL), and baseline + distance loss + SE block (SO-DLSE) (the structure can be seen in Figure 5). It should be noted that since our baseline network is based on the model from [31], we trained and tested the experimental results from scratch with our own BabyExp dataset for better comparison. Same as in [25], here, we use the center loss [32] in any case to train the network, not the locality preserving loss [31], because we do not deal with compound emotions. Table 4 shows the results of this experiment. In order to objectively measure the performance, the BabyExp dataset is divided into training and test sets, where the test set contains 2,413 images, and the remaining 9,640 images are used as the training set. The dataset is then resized to a fixed size 100×100, which is subsequently sent to the CNN classifier for expression recognition. It should be noted that the image size is resized to 224×224 only when entering the VFESO-DLSE method. The labeled facial expression dataset is quite small; thus, we use the conventional data augmentation method to generate more training data. In the data augmentation stage, we augment the set of training images in BabyExp by random flipping, rotating each with $\pm 10^{\circ}$, and random crop. We then train our networks for 700 epochs with the following parameters: learning rate 0.0001–0.005, weight decay 0.05, momentum 0.9, batch size 128, and linear learning rate decay in the

Table 3: Experimental results of adult expression recognition models on the adult and BabyExp dataset.

Models	SFEW2.0	BabyExp
DLP (trained on SFEW2.0)	54.45	39.70
Baseline (trained on SFEW2.0)	58.14	40.78

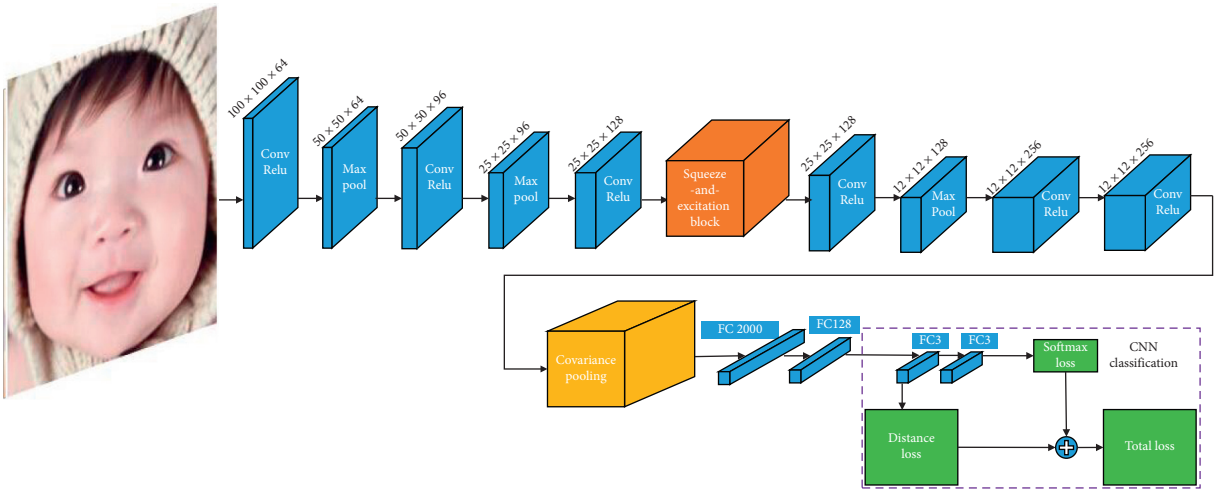


Figure 5: The overall architecture of the deep learning approach SO-DLSE.

Table 4: The expression recognition performance of different methods on the BabyExp dataset (trained from scratch).

Models	Happy			Sad			Normal			Average accuracy
	Recall	Precision	F1	Recall	Precision	F1	Recall	Precision	F1	
DLP [31]	62.25	66.76	64.42	53.04	82.96	64.90	79.70	56.20	65.92	65.02
Baseline [25]	65.13	80.53	72.01	82.24	88.86	85.42	91.21	72.18	80.59	79.57
SO-DL	52.75	92.34	67.14	93.91	85.52	89.52	97.03	73.13	83.40	81.31
SO-DLSE	59.62	91.73	72.27	93.54	89.22	91.33	96.04	73.98	83.58	83.13
VFESO-DLSE	78.5	93.18	85.21	96.27	85.07	90.33	88.86	86.71	87.78	87.90

Adaptive Moment Estimation (Adam) optimizer. It is worth pointing out that, to better measure the availability of the BabyExp dataset and the accuracy of the results, we report total accuracy, per class precision, per class recall, and per class F1-measure as the evaluation metrics here.

The last experiment is to verify the experimental results if the data are not equalized by CycleGAN. Table 5 shows the results of this experiment. The original dataset contains 7,600 pictures, including 2,502 happy images, 4,028 normal images, and 1,070 sad images. In order to objectively measure the performance, it is divided into training and test sets. The test set contains 1,522 images, and the remaining 6,078 images are used as the training set. We choose two methods with better experimental results in the second experiment: SO-DLSE and VFESO-DLSE. Experimental settings, parameter settings, and the number of iterations are the same as those in the second experiment above.

3. Results

Table 3 shows the experimental results of adult expression recognition models trained on the adult dataset and tested on the adult and BabyExp datasets. As we can see, the performance of these methods on the BabyExp is significantly lower than that on the adult dataset SFEW2.0, 54.45% on SFEW2.0 vs. 39.7% on BabyExp and 58.14% on SFEW2.0 vs. 40.78% on BabyExp, indicating that baby faces are greatly different from the adult faces, and it is important for developing facial expression recognition approaches for baby images.

The overall expression recognition performance of the proposed different experiments trained from scratch on the BabyExp dataset is shown in Table 4. From the results, we have the following observations: firstly, we can clearly see that the accuracy of DLP and baseline methods when trained and tested from scratch on the BabyExp dataset has greatly improved, 39.7% to 65.02% and 40.78% to 79.57%, compared with that trained on adult dataset SFEW2.0, once again indicating that baby faces are greatly different from the adult faces. Secondly, our proposed method VFESO-DLSE achieves the best result, 87.90%, which is about 4.8% greater than SO-DLSE showing that VGG16 is better than other CNN methods to extract features. From the results of baseline, SO-DL, and SO-DLSE, we can see distance loss and SE can achieve an improvement about 1.8%. The purpose of the distance loss is to learn lower changes between the same classes and higher distances between different classes, and the SE block can automatically obtain the importance of each feature channel through learning. Thirdly, from the results, it

Table 5: The expression recognition performance of original data which are not equalized by CycleGAN (trained from scratch).

Models	Happy			Sad			Normal			Average accuracy
	Recall	Precision	F1	Recall	Precision	F1	Recall	Precision	F1	
SO-DLSE	53.60	47.35	50.28	0.0	0.0	0.0	77.23	65.27	70.75	58.61
VFESO-DLSE	56.40	84.18	67.54	38.79	76.14	51.39	94.68	70.96	81.12	74.24

is obviously shown that the recall, precision, and F1-measure can further confirm the reliability of our results and the validity of our method.

The expression recognition performance of original data which are not equalized by CycleGAN can be seen in Table 5. We have two observations of the facial expression recognition on BabyExp. Firstly, we can easily see that two methods, SO-DLSE and VFESO-DLSE, have achieved 58.61% and 74.24% on the original data, in which both are still lower than 83.13% and 87.90% on BabyExp equalized by CycleGAN from Table 4. Secondly, even though these two methods have achieved higher accuracy, the recall rate and F1-measure are not very high, especially for the sad expression; this is because the distribution of expressions is unbalanced, and models trained using imbalance original facial expression samples have poor generalization ability and are prone to overfitting. Even in the SO-DLSE method, the recall, precision, and F1-score values of sad expressions are all 0, while the VFESO-DLSE method obtained 38.79%, 76.14%, and 51.39% in recall, precision, and F1-score, respectively, which also shows on the one hand that VGG16 is better than other CNN methods to extract features. On the other hand, it shows that we need to perform data augmentation to promote data balance and facilitate the use of deep learning methods for experiments, which validates the importance of CycleGAN for data equalization. This conclusion can also be drawn from the experimental results in Table 4.

4. Discussion

Facial expression recognition (FER) has always been a challenging topic in computer vision. Researchers usually aim to build a system that can identify different expressions in the images automatically [33]. Research on facial expression recognition relies heavily on an adequate dataset of facial expressions. However, due to the inherent nature of facial expressions and the difficulty of obtaining them, there are currently only a limited number of publicly available databases, which provide a sufficient number of facial images and are tagged with accurate facial expression information. Table 1 shows the summary of the existing image databases with the number of images, number of subjects, expression distribution, data size, and released years.

However, there are several limitations for these datasets. Most of the existing works and datasets [7, 8] focus on analyzing adult faces, which ignore how to analyze facial expressions from baby facial images. Recently, researchers have realized the importance of children facial expressions in order to study developmentally the interpretation of these expression datasets. For example, the new NIMH Children's Emotional Face Picture Collection (NIMH-ChEFS) contains photos of children aged 10–17 [12], the Radboud Faces Database includes photos of 8- to 12-year-olds [13], and the CAFE set features photographs of 2- to 8-year-old children [14]. Although these new datasets give researchers the option to use a sample of children aged 2–17 years, there have been no datasets that include younger children to date. On the contrary, all the datasets mentioned above for children facial expressions have only a small number of images, which are not suitable for training CNN models. In addition, these datasets contain posed expressions in the lab-controlled environment, not spontaneous or natural facial expressions.

5. Conclusions

In this paper, to address the aforementioned issues, we propose a new image dataset with expression labels of baby faces for automatic facial expression recognition. Our dataset, which we call the BabyExp dataset, contains more than 12,000 images from babies younger than two years old showing spontaneous expressions in an uncontrolled environment. Each face image is annotated with one of three facial expressions (i.e., happy, sad, and normal). It is complementary to the existing adult face dataset and can shed some light on exploring baby face analysis, and it will enable the academic research community to study baby faces in a manner comparable to the vast literature that relies heavily on adult faces.

As a result, our novel dataset will become an important milestone for human expression researchers. This dataset will be an important resource for the computer vision community to benchmark and compare results. We further evaluate state-of-the-art adult face analysis methods on BabyExp, which indicate that adult facial expression recognition methods are not suitable for baby facial expression recognition, and new methods are necessary to be developed to approach baby face recognition. Besides, we have also proposed a baseline for automatic expression recognition for babies based on deep learning. We conduct several experiments and report the baseline performances of the BabyExp dataset. The proposed baseline CNN architecture achieves an average classification accuracy of 87.90% on the BabyExp dataset. The performance of these methods on the BabyExp dataset is significantly lower than that on the other datasets, indicating that baby face facial images are greatly different from the adult faces, and it is important for the community to develop facial expression recognition approaches for babies.

We hope that the release of the BabyExp dataset will encourage more research works on the real-world children expression recognition, and it will be a useful benchmark

resource for researchers to validate their facial expression analysis algorithms in challenge conditions. We will collect more data and assign more specific facial expression labels (i.e., crying and laughing) to each image in order to extend the dataset. And we will continue to explore methods to achieve better performance for baby facial expression recognition in the future.

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] M. Pantic and L. J. M. Rothkrantz, "Automatic analysis of facial expressions: the state of the art," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1424–1445, 2000.

[2] A. Di Domenico, R. Palumbo, N. Mammarella, and B. Fairfield, "Aging and emotional expressions: is there a positivity bias during dynamic emotion recognition?" *Frontiers in Psychology*, vol. 6, p. 1130, 2015.

[3] M. Altamura, F. A. Padalino, E. Stella et al., "Facial emotion recognition in bipolar disorder and healthy aging," *The Journal of Nervous and Mental Disease*, vol. 204, no. 3, pp. 188–193, 2016.

[4] R. Palumbo, R. B. Adams Jr, U. Hess, R. E. Kleck, and L. Zebrowitz, "Age and gender differences in facial attractiveness, but not emotion resemblance, contribute to age and gender stereotypes," *Frontiers in Psychology*, vol. 8, p. 1704, 2017.

[5] H. Ding, S. K. Zhou, and R. Chellappa, "Facenet2expnet: regularizing a deep face recognition net for expression recognition," in *Proceedings of the 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pp. 118–126, IEEE, Washington, DC, USA, June 2017.

[6] Y. Fan, J. C. Lam, and V. O. K. Li, "Multi-region ensemble convolutional neural network for facial expression recognition," in *Proceedings of the 27th International Conference on Artificial Neural Networks*, pp. 84–94, Springer, Rhodes, Greece, October 2018.

[7] A. Dhall, R. Goecke, S. Lucey, and T. Gedeon, "Static facial expression analysis in tough conditions: data, evaluation protocol and benchmark," in *Proceedings of the 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 2106–2112, IEEE, Barcelona, Spain, November 2011.

[8] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, "The extended cohn-kanade dataset (ck+): a complete dataset for action unit and emotion-specified expression," in *Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*, pp. 94–101, IEEE, San Francisco, CA, USA, August 2010.

[9] H. Yang, Z. Zhang, and L. Yin, "Identity-adaptive facial expression recognition through expression regeneration using conditional generative adversarial networks," in *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pp. 294–301, IEEE, Xi'an, China, June 2018.

[10] C. M. Kuo, S. H. Lai, and M. Sarkis, "A compact deep learning model for robust facial expression recognition," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 2121–2129, Salt Lake City, UT, USA, December 2018.

[11] P. D. M. Fernandez, F. A. G. Peña, T. I. Ren, and A. Cunha, "FERAtt: facial expression recognition with attention net," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Long Beach, CA, USA, June 2019.

[12] H. L. Egger, D. S. Pine, E. Nelson et al., "The NIMH child emotional faces picture set (NIMH-ChEFS): a new set of childrens̈ facial emotion stimuli," *International Journal of Methods in Psychiatric Research*, vol. 20, no. 3, pp. 145–156, 2011.

[13] O. Langner, R. Dotsch, G. Bijlstra, D. H. J. Wigboldus, S. T. Hawk, and A. Van Knippenberg, "Presentation and validation of the radboud faces database," *Cognition & Emotion*, vol. 24, no. 8, pp. 1377–1388, 2010.

[14] V. LoBue and C. Thrasher, "The child affective facial expression (CAFE) set: validity and reliability from untrained adults," *Frontiers in Psychology*, vol. 5, p. 1532, 2015.

[15] D. E. King, "A machine learning toolkit," *Journal of Machine Learning Research*, vol. 10, no. 60, pp. 1755–1758, 2009.

[16] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[17] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., "Generative adversarial nets," *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

[18] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2223–2232, Venice, Italy, October 2017.

[19] M. Xia, X. Zhang, W. a. Liu, L. Weng, and Y. Xu, "Multi-stage feature constraints learning for age estimation," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2417–2428, 2020.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, no. 2, pp. 1097–1105, 2012.

[21] L. Weng, L. Wang, M. Xia, H. Shen, J. Liu, and Y. Xu, "Desert classification based on a multi-scale residual network with an attention mechanism," *Geosciences Journal*, pp. 1–13, 2020.

[22] M. Xia, W. Liu, Y. Xu, K. Wang, and X. Zhang, "Dilated residual attention network for load disaggregation," *Neural Computing and Applications*, vol. 31, no. 12, pp. 8931–8953, 2019.

[23] M. Xia, J. Qian, X. Zhang, J. Liu, and Y. Xu, "River segmentation based on separable attention residual network," *Journal of Applied Remote Sensing*, vol. 14, no. 3, Article ID 032602, 2019.

[24] K. Yu and M. Salzmann, "Second-order convolutional neural networks," 2017, http://arxiv.org/abs/1703.06817.

[25] D. Acharya, Z. Huang, D. Pani Paudel, and L. Van Gool, "Covariance pooling for facial expression recognition," in *Proceedings of the 2018 The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPRW)*, Salt Lake City, UT, USA, June 2018.

[26] J. Donahue, Y. Jia, O. Vinyals et al., "Decaf: a deep convolutional activation feature for generic visual recognition," *Proceedings of the 31st International Conference on Machine Learning*, vol. 32, no. 1, pp. 647–655, 2014.

[27] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: an astounding baseline for recognition," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 806–813, Columbus, OH, USA, June 2014.

[28] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, http://arxiv.org/abs/1409.1556v6.

[29] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, Salt Lake City, UT, USA, June 2018.

[30] M. Abadi, A. Agarwal, P. Barham et al., "Large-scale machine learning on heterogeneous distributed systems," 2016, http://arxiv.org/abs/1603.04467.

[31] S. Li and W. Deng, "Reliable crowdsourcing and deep locality-preserving learning for unconstrained facial expression recognition," *IEEE Transactions on Image Processing*, vol. 28, no. 1, pp. 356–370, 2018.

[32] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Proceedings of the European Conference on Computer Vision*, pp. 499–515, Springer, Amsterdam, The Netherlands, October 2016.

[33] S. Xie, H. Hu, and Y. Wu, "Deep multi-path convolutional neural network joint with salient region attention for facial expression recognition," *Pattern Recognition*, vol. 92, pp. 177–191, 2019.

Hindawi
Complexity
Volume 2020, Article ID 8706285, 14 pages
https://doi.org/10.1155/2020/8706285

Research Article

Improved ML-Based Technique for Credit Card Scoring in Internet Financial Risk Control

Shuangshuang Fan,[1] **Yanbo Shen**,[2] **and Shengnan Peng**[1]

[1]*School of Management, China University of Mining and Technology-Beijing, Beijing, CO 100080, China*
[2]*Dahua Certified Public Accountants, Beijing, CO 100080, China*

Correspondence should be addressed to Shuangshuang Fan; 934042440@qq.com

Received 17 July 2020; Revised 16 September 2020; Accepted 17 October 2020; Published 4 November 2020

Academic Editor: Min Xia

With the rapid development of China's Internet finance industry and the continuous growth of transaction amount in recent years, a variety of financial risks have increased, especially credit risk in the financial industry. Also, the credit risk evaluation is usually made by using the application card scoring model, which has the shortcomings of strict data assumption and inability to process complex data. In order to overcome the limitations of the credit card scoring model and evaluate credit risk better, this paper proposes a credit evaluation model based on extreme gradient boosting tree (XGBoost) machine learning (ML) algorithm to construct a credit risk assessment model for Internet financial institutions. At the same time, an Internet lending company in China is taken as a case study to compare the performance of the traditional credit card scoring model and the proposed machine learning (ML) algorithm model. The results show that ML algorithm has a very significant advantage in the field of Internet financial risk control, it has more accurate prediction results and has no particularly strict assumptions and restrictions on data, and the process of processing data is more convenient and reliable. We should increase the application of ML in the field of financial risk control. The value of this paper lies in enriching the related research of financial technology and providing a new reference for the practice of financial risk control.

1. Introduction

Since the 1970s, human society has entered the industrial 3.0 era marked by the application of electronic information technology, and computer technology and Internet have been widely used in various fields and integrated with traditional industries, giving birth to new business models and formats [1]. With the rapid development of China's economy and the popularity of network technology, the traditional financial industry and Internet technology are integrated and derived into a series of network-based financial products [2]. However, due to the imperfection of the trading system and the lack of convenience of operation, Internet finance did not enter the public's attention until "Yu'E Bao" was launched by a financial service company in 2013, leading to the vigorous development stage of Internet finance [3]. Relying on big data and cloud computing technology, Internet finance forms functional financial formats and services in the open Internet platform, including Internet innovation and e-commerce innovation of traditional financial institutions, APP software, e-commerce enterprises of nonfinancial institutions using Internet technology for financial operation, P2P network credit platform, crowd-funding network investment platform, and financial resource mode of mobile financing APP and third-party payment platform [4]. At present, Internet finance has been on a healthy development track in the strategic environment of "green finance" and "science and technology power" advocated by the state and the Chinese government [5].

Due to the late start of China's Internet finance, the regulatory system needs to be improved. Internet finance not only brings vitality to financial enterprises and social financing and investment activities but also causes various

potential risks and challenges. From 2016 to 2018, more than 200 Internet financial platforms in China have defaulted. The credit risk of the involved platforms leads to huge losses, such as operators' fraud or loss of money with them, overdue repayment by borrowers, and collapse of P2P platforms [6]. Based on the increasing negative effect of Internet financial risk on society, it is urgent to establish an effective risk control system. In the traditional financial industry, the credit scoring card model is usually established to deal with credit risk. It uses a large number of historical credit data to describe the customer's income status, credit history, payment level, and other indicators and gives different weights. The indicators are divided into several levels and scored according to the historical data of customers to obtain the relevant credit rating [7].

However, due to the complexity of the modelling process and the limited accuracy of processing a large number of highly complex information, the traditional credit score card model is prone to bias and has some limitations in Internet financial risk management [8]. In this paper, the ML model is proposed to predict credit risk by collecting and mining Internet data, repeatedly calculating, and verifying. Through case study and empirical study, it is concluded that under the same data sources, the ML model has higher accuracy and recall rate than the traditional credit scoring model, and it plays an important role in the Internet financial risk control system. The main contributions of this paper are as follows.

1.1. Contribution in Theory. This paper enriches the theory of ML in the field of financial risk control. The theoretical research on the application of ML in Internet financial risk control in China has not yet formed a perfect system. At the same time, most of the foreign research focuses on the financial market system risk early warning, anti-money laundering of financial institutions, and other aspects, and research focusing on the content of Internet financial risk control is relatively small. In this paper, the application of ML algorithm in the credit risk management of Internet finance is first proposed, which has strong innovation.

1.2. Contribution in Practice. In view of the serious credit risk in China's Internet financial industry, this paper proposes a financial risk control method based on ML algorithm. At the same time, the case study verifies the superiority of the proposed method. Therefore, this study provides valuable and meaningful guidance for the risk management of the actual Internet financial industry and helps to reduce the risk of China's Internet financial industry.

This paper proposes that universities, scientific research institutions, and Internet financial industry should cooperate and communicate with each other. It promotes the latest research results in ML algorithm of scientific research institutions, which can well transfer the value of its practice, that is, to serve the Internet financial industry. The application of science and technology is emphasized. It promotes the close relationship between industry and academia, thus contributing to the strategy of "rejuvenating the country through science and education" advocated by China.

1.3. Contribution for Further Research. This paper presents the application of ML algorithm in Internet financial risk control. Because the traditional risk assessment method has been widely used and has strong interpreted ability, the ideal situation is that the two methods are effectively combined. Then, the proposed method provides a preliminary reference for the future combination of traditional credit scoring model and ML algorithm model. With the deepening of future research, we will explore how to effectively combine different advanced methods.

The remainder of this paper is organized as follows.

Section 2 gives the background and related work. This section reviews the related research results of financial industry risk control at home and abroad and points out the shortcomings of these achievements and the basic ideas of this paper. Section 3 presents the technical model. This section describes the credit card scoring theory model and XGBoost ML method model and points out the evaluation model-related indicators. Section 4 is devoted to case study and empirical analysis. We take a P2P enterprise in China as an example and analyze the advantages of the proposed model. In Section 5, we draw a conclusion.

2. Background and Related Work

This section will systematically introduce the research background and related work. It lists the relevant research on the credit evaluation method by international scholars and Chinese scholars through reading literature. The specific work in this section is divided into academic research on credit scoring, international scholars' research on ML in the field of financial risk control, and Chinese scholars' research on financial risk control.

At the end of this section, it is pointed out that the traditional credit scoring model for financial risk control has limitations, that is, the data have strict assumptions, and it must be linear and cannot process large-scale data. The main research content of this paper is the application of ML algorithm in the field of Internet financial risk control.

2.1. Traditional Credit Scoring Model. The history of credit scoring in the world can be traced back to the 1950s. Mathematician Earl Isaac and engineer Bill Fair first established the world's first commercial credit scoring system FICO and extended it to the financial system [9]. After that, financial institutions make credit decisions through the 5C credit discrimination method [10]. 5C discriminant analysis is composed of five evaluation factors, such as the lender's role, capital, collateral, capacity, and environment. It comprehensively forecasts the performance of borrowers. The limitation of this method is that it is inefficient in processing large-scale data.

However, with the expansion of the loan scale of financial institutions and the increase of the number of borrowers, the above credit evaluation method is not

applicable, and a new method, that is, the credit scoring method, has been adopted. Financial institutions build data-driven models based on quantifiable characteristics of borrowers to manage credit risk [11]. The credit score is based on the historical credit data of the borrower, and the credit score is calculated by the model, and the credit granting person determines whether to grant credit or not and the credit line according to the credit score. Hand and Henley pointed out that the statistical techniques and quantitative methods in the construction of scorecards have been extended from discriminant analysis and linear regression methods widely used in the early stage to logical regression, probit regression, nonparametric smoothing method, Markov chain model, recursive segmentation, expert system, and genetic algorithm [12].

Subsequently, Lee and other scholars empirically studied the effectiveness of using multiple adaptive regression spine (MARS) and classified regression tree (CART) for credit scoring. The two methods are superior to the traditional discriminant analysis and logical regression methods in the accuracy of credit scoring [13]. According to Bee et al. [14], with the development of current data mining technology, the process of establishing a credit scoring model is more convenient, and various new technologies have been developed. However, in the practical application of financial institutions, the commonly used technologies are still logical regression and decision tree because such technologies are more convenient in identifying important input variables, interpreting results, and building models.

2.2. Application of ML in Financial Risk Control. With the development of big data and data mining technology, international scholars have formed rich research results on ML in credit risk prediction and evaluation. Because the goal of the credit management of financial institutions is to optimize the business performance and minimize the risk, decision rules should be established to make credit decisions. Therefore, clustering algorithm is widely used in the credit scoring system in the early stage. For example, William and Huang combined the K-means clustering method with the supervision method for insurance risk identification [15].

Furthermore, Yeo et al. [16], used hierarchical clustering technology to predict the risk of automobile insurance industry. Different customer risk levels are identified by clustering technology to make operational decisions on credit limit. With the increase of data scale, scholars try to build more complex models, such as Khandani, Kim and so on. They use ML algorithm and statistical model to predict consumer default risk with massive customer transaction records and credit management agency data. Their research results show that ML technology reduces the prediction error of 6% to 25% compared with the traditional linear regression model [17]. Chakrabort and Joseph trained a set of financial distress prediction model based on ML and proposed that the ML method was better than the statistical models such as logical regression. In terms of the receiver operating characteristic area discrimination, there is about 10% significant improvement [18]. Ticknor proposed to use neural network algorithm to predict financial market behavior. Empirical results show that the model constructed by this algorithm has the same prediction effect as the advanced model without data preprocessing [19]. Gogas and Agrapetidou constructed a prediction model of financial institutions bankruptcy based on support vector machine, analyzed the data of financial statements publicly disclosed by banks, and predicted the number of bankruptcies of American financial institutions from 2007 to 2013. The model shows 99.2% prediction accuracy [20].

Rtayli and Enneya proposed an enhanced credit card risk identification method based on random forest classifier and support vector machine feature selection algorithm to predict fraud risk. Experimental results show that the classification performance of the algorithm is better than that of local outlier factor, isolated forest, and decision tree algorithms on large datasets [21]. Plawiak et al. proposed deep genetic hierarchical learner network (DGHLN) algorithm, which is an excellent learner training method based on genetic hierarchical training. 21% of the credit rate in Germany was verified by cross validation [22].

2.3. Current Situation and Background of Related Research in China. China's credit reporting system has not yet entered a mature stage. At present, less than 50% of the population in China can generate credit report in the People's Bank of China, which limits the accuracy of the traditional credit scoring card model in assessing the credit risk of the lender. With the advent of the Internet era in recent years, big data and artificial intelligence technologies have gradually developed and spread in domestic financial market risk control, making up for the lack of credit data. By analyzing the borrower's Internet information and converting it into feature vector, ML algorithm is used to predict the potential default risk. The success of this model in Internet financial risk control has attracted domestic scholars' in-depth research. For example, Hou and Liu applied the support vector machine nonlinear classifier to the bank credit risk assessment and analyzed and compared the experimental results with different kernel functions and parameters [23].

Subsequently, Hou and Xue used the approximate support vector machine (PSVM) model in ML principle to conduct an empirical analysis on the personal housing loan data of a commercial bank in Xi'an market. The results show that the accuracy of the model in predicting the credit risk of individual housing loans of commercial banks reaches 87.5% [24]. Hu et al. established the credit risk assessment model under the supply chain finance mode by using support vector machine. By comparing with the model established by principal component analysis and logical regression method, it is confirmed that the credit risk assessment system based on SVM is more effective and superior [25].

Based on the idea of data mining, Zhao and Chen used customer credit consumption behavior data and rough set theory to reduce the condition attributes in the decision table, constructed a decision tree algorithm based on variable precision weighted average roughness and Gini index, and predicted the default repayment of customers according

to the decision attribute value. The experimental results show that the improved dynamic early warning model of credit card consumption credit risk based on rough set and decision tree algorithm is often better than the basic statistical model and ML algorithm in terms of accuracy and stability [26]. Liu and Tang used the area under the ROC curve AUC value as the classification performance index of the binary classification algorithm, constructed a feature selection algorithm AUCRF based on random forest algorithm, and made an empirical analysis of Australian credit data in UCI ML database. The results show that the model based on AUCRF algorithm can obtain higher classification performance with smaller feature subset, AUC = 0.9346 [27].

2.4. ML in Credit Scoring for Internet Financial Risk Management. Note that many methods and technologies for risk control in the financial field have been proposed in the existing literature, including traditional methods for credit risk management in Internet finance. However, this paper first mainly introduces ML algorithm into Internet finance credit risk management. We can verify the innovation of this paper by comparing the existing research results of credit risk management with the contents of this paper.

This study uses "Internet financial credit scoring," "ML in Credit Scoring," and "application ML and Internet financial risk control" as keywords to search. The search scope is review articles on financial risk management published from 2010 to 2020. The study selected peer-reviewed journals and conference articles because of their high quality. We choose the article by reading the conclusion and abstract, and sometimes we need to read the whole article. All unpublished work and dissertations are not included in this current study. Other existing literatures include systematic research on bankruptcy forecasting or the use of credit scoring models, as well as the application of ML in traditional financial field. Table 1 lists the literature investigated and does not mention the application of ML algorithm in the field of Internet financial risk management.

Through literature review, it can be seen that focusing on the research of ML algorithm applied in traditional credit scoring model in the field of Internet financial credit risk management research is insufficient. However, the traditional credit evaluation methods have limitations in multidimensional and large-scale data analysis, and the model method has strict limitations in distribution hypothesis and linearity. It is difficult for Internet credit data to meet the requirements of the traditional model. The ML algorithm based on big data and artificial intelligence can make accurate analysis and prediction of multisource and multitype data and has developed rapidly. Traditional risk measurement methods predict the future default risk based on the borrower's historical data and personal characteristics, while ML algorithm has extensive expansion in the dimension of obtaining information, which can deeply analyze the correlation between such information and default risk based on behavioral information, soft information, and hard information. In the current research, there are few literatures comparing the traditional credit evaluation model and ML model, and the research on the integration of the two methods to evaluate the credit risk of Internet finance is relatively rare. Therefore, on the basis of reading the relevant literature at home and abroad, this paper uses ML algorithm to construct the credit risk model, verifies the performance of ML model better than the traditional credit score card model through empirical verification, makes a deep discussion on how to convert the ML model into the score card model, and puts forward suggestions on the construction of risk control system of Internet financial industry by ML.

3. Model and Evaluation Metric

In this part, the algorithms of credit scoring model and ML model will be discussed. In addition, some evaluation indexes about the performance of the model are introduced. The function of this part is to lay the foundation for the case study and empirical analysis in the next section.

3.1. Credit Scoring Model. Credit scoring is a supervised learning method, which is essentially a binary classification. According to the historical data characteristics of customers of various categories, a mathematical model is established to predict the default risk of lenders according to "good borrowers" and "bad borrowers" [36]. Because of its strong interpreting ability, logistic regression (LR) is the most commonly used model in credit scoring. The formula of logistic regression model is as follows:

$$P(y=-1|x) = 1 - P(y=+1|x) = \frac{\exp\left(a_0 + a^T x\right)}{1+\exp\left(a_0 + a^T x\right)},$$
$$P(y=+1|x) = \frac{1}{1+\exp\left(a_0 + a^T x\right)}, \tag{1}$$

where $x \in R$ is feature vector; $p(y=+1|x)$ is the probability that the eigenvector borrower x is classified as a non-defaulting customer; and $p(y=-1|x)$ is the probability that the eigenvector borrower x is classified as a defaulting customer. $\{a_0, a\}$ represents whether the model parameters are estimated by using, for example, the maximum likelihood estimation of the training dataset [37]. Once the model parameters are estimated, the decision on the eigenvector x is recorded as $\hat{y} = +1, \quad \text{if}$

$$P(y=+1|x) \geq P(y=-1|x). \tag{2}$$

According to the above calculation of customer credit evaluation process, credit decision rules can be summarized as follows:

$$\hat{y} = \begin{cases} +1, & \text{for } 1 \geq \exp\left(a_0 + a^T x\right), \\ -1, & \text{otherwise.} \end{cases} \tag{3}$$

3.2. XGBoost Integrated Learning Method. Qi de et al. proposed the XGBoost algorithm [38] solving real-world

Table 1: Existing literature surveys on financial risk management and their differences from this survey paper.

Survey paper	Articles searched	Objective	Difference from this paper
[28]	165	Research on the bankruptcy risk of financial institutions Investigation of model type by decade Compare model performance	Internet financial risk management was not mentioned The application of ML algorithm was not involved Credit risk was not considered
[29]	214	Research on the application of traditional credit card scoring model Comparing models based on performance	Deep learning models were not covered
[30]	130	ML in financial crisis prediction Focus on private enterprise	Credit risk was not covered Not related to financial industry
[31]	Not specified	Comparing models based on credit rating	Internet financial risk management was not mentioned Lack of evidence to prove that it is advisable to introduce ML algorithm into credit card scoring model
[32]	187	Proposing a new method for scoring Compare traditional techniques Conceptual discussion	Deep learning models were not covered Internet financial risk management was not included
[33]	6	Focus on bankruptcy prediction Models are compared based on design, datasets, and baselines	Internet financial risk management was not included Credit risk was not covered
[34, 35]	49	The search for models to predict the prices of financial markets	Internet financial risk management was not included Credit risk was not covered

classification problem. They posit that XGBoost is an optimized version of gradient boosting machine. The main improvement on GBDT is the normalization of the loss function to mitigate model variances. This also reduces the complexities of modelling and hence the likelihood of model overfitting [39]. Meanwhile, the conventional method uses decision trees as a classification basis. In contrast, XGBoost supports linear classifiers, applicable not only to classifications but also to linear regressions. The traditional approach only deals with the first derivative in learning but XGBoost improves the loss function with Taylor expansion. While the level of complexities increases for the learning of trees, the normalization prevents the problems associated with overfitting [40].

The algorithm has unique advantages in sparse data processing, approximate tree building, and parallel computing, which makes ML technology widely used in mechanical engineering, rail transit, automation technology, and other fields [41]. XGBoost is a gradient lifting ensemble algorithm based on decision tree and linear model. Its basic idea is to combine some decision tree models to form a model with high accuracy. If we give the data as $(x_1, y_1), (x_2, y_2), \ldots, (x_i, y_i), \quad i = 1, 2, \ldots, n$, x_i represents the independent variable, and y_i represents the a dependent variable. The calculation steps are as follows:

$$\hat{y}_i^t = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i), \tag{4}$$

where $\hat{y}_i^t$ is the predicted value of the model in the round t and XGBoost model algorithm is formed by continuous iteration, and each iteration is trained by adding a lesson of decision tree to the prediction value $\hat{y}_i^t$ of the previous round. In general, the formula of the objective function is as follows:

$$\text{obj}(w) = L(w) + \Omega(w), \tag{5}$$

where w is the parameter to be estimated, $L(w)$ is the loss function, and $\Omega(w)$ is the regularization term. Therefore, minimizing $\text{obj}(w)$ is the criterion for selecting $f(x)$.

$$\begin{aligned}\text{obj}^{(t)} &= \sum_{i=1}^{n} l\left(y_i, \hat{y}^{(t)}\right) + \sum_{i=1}^{t} \Omega(f_i) \\ &= \sum_{i=1}^{n} l\left(y_i, \hat{y}^{(t-1)} + f(x)\right) + \Omega(f_i) + \text{constant}.\end{aligned} \tag{6}$$

Taylor expansion is used to expand the approximate objective function and remove the constant term. The final objective function is as follows:

$$\begin{aligned} g_i &= \partial_{\hat{y}(t-1)} l\left(y_i, \hat{y}^{(t-1)}\right), \\ h_i &= \partial^2_{\hat{y}(t-1)} l\left(y_i, \hat{y}^{(t-1)}\right), \\ \text{obj}^{(t)} &= \sum_{t=1}^{n}\left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)\right] + \Omega(f)_t. \end{aligned} \tag{7}$$

In XGBoost algorithm, the following improvements will be made: the decision tree is divided into the structure part Q of the tree and the weight (fraction) part w of the leaf node.

$$f_t(x) = w_{q(x)}. \tag{8}$$

Moreover, the complexity of the tree is redefined as

$$\Omega(f)_t = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2, \tag{9}$$

where T represents the number of leaf nodes. Under these new definitions, the new form of objective function is

$$\begin{aligned}\mathrm{obj}^{(t)} &= \sum_{t=1}^{n}\left[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)\right] + \Omega(f)_t,\\ &= \sum_{t=1}^{n}\left[g_i w_{q(x)} + \frac{1}{2}h_i w_q^2(x)\right] + \gamma T + \frac{1}{2}\lambda\sum_{j=1}^{T} w_j^2,\\ &= \sum_{j=1}^{T}\left[\left(\sum_{i\in I_j} g_i w_j + \frac{1}{2}\left(\left(\sum_{i\in I_j} h_i\right) + \lambda\right)\right)w_j^2\right] + \gamma T.\end{aligned} \tag{10}$$

If $G_j = \sum_{i\in I_j} g_i$, , $H_j \sum_{i\in I_j} h_i$, the objective function can be further rewritten as

$$\mathrm{obj}^{(t)} = \sum_{j=1}^{T}\left[G_i w_j + \frac{2}{2}(H_i + \lambda)w_j^2\right] + \gamma T. \tag{11}$$

After the objective function is obtained, the optimal value of w_j can be obtained by finding the reciprocal of w_j and making it equal to zero:

$$w_j^* = -\frac{G_j}{H_j + \lambda}. \tag{12}$$

Substituting equation (12) into the objective function, we can get

$$\mathrm{obj} = -\frac{1}{2}\sum_{j=1}^{T}\frac{G_j^2}{H_j + \lambda} + \gamma T. \tag{13}$$

3.3. Evaluation Metric. For the traditional credit scoring model, in order to improve the speed and accuracy of calculation, we need to select variables before establishing the model. The choice of variables is based on their information value, which is abbreviated as IV. Information value describes the importance of the contribution of variables to the prediction results of the model. We choose to add variables with high IV value to the model, while variables with too small IV value will not be added to the model. If we want to calculate IV, firstly need to calculate the WOE, that is, the weight of evidence. WOE is a form of encoding the original independent variable. If you want to code a variable, you need to first group the variable (also known as discretization, boxing, etc.); after grouping, the calculation formula of WOE for group I is as follows:

$$\begin{aligned}\mathrm{WOE}_i &= \ln\left(\frac{py_i}{pn_i}\right)\\ \mathrm{IV}_i &= (py_i - pn_i)\times \mathrm{WOE}_i,\\ &= (py_i - pn_i)\times \ln\left(\frac{py_i}{pn_i}\right),\\ &= \left(\frac{y_i}{y_T} - \frac{n_i}{n_T}\right)\times\left(\frac{y_i/y_T}{n_i/n_T}\right),\\ \mathrm{IV} &= \sum_{i}^{n}\mathrm{IV}_i,\end{aligned} \tag{14}$$

where Py_i is the proportion of bad samples to all bad samples in this group, Pn_i is the proportion of good samples to all good samples in this group, y_i is the number of bad samples in this group, n_i is the number of good samples in this group, y_T is the number of all good samples in the sample, and n_T is the number of all bad samples in the sample.

For the machine learning model, there are many evaluation indexes, and the commonly used indexes are accuracy rate, true positive rate, false positive rate, accuracy rate, F1 score, etc., which are shown in Table 2. We can also construct confusion matrix based on these indicators, which is shown in Table 3. In addition, we also draw the receiver operating characteristic (ROC) curve and the Kolmogorov–Smirnov (KS) curve of the subjects to reflect the performance of the model more vividly.

AR measures the overall predictive effectiveness of model; however, it is not a reliable parameter as it yields misleading results if the dataset is not balanced. The parameters mentioned above are calculated based on the confusion matrix shown in Table 1. True positive (TP) refers the number of defaults that are correctly predicted as defaults; false positive (FP) refers the number of nondefaults that are mistakenly predicted as defaults; true negative (TN) refers the number of nondefaults that are correctly predicted as nondefault; false negative (FN) refers the number of defaults that are mistakenly predicted as nondefaults. In addition to these evaluation indexes, there are two very important ML model prediction performance indicators, such as AUC and KS curve.

3.3.1. ROC and AUC. When the output of the model classifier is continuous, the AUC value can be used as the evaluation standard, and its value range is AUC∈[0, 1]. If we use f to represent a classifier, "$x\_$ "to represent negative samples and" X_+" to represent positive samples, the output result of f is $(x\_) < f(x_+)$, the ROC curve of the classifier passes through the point (0, 1), and the corresponding AUC value is 1. The AUC value of the normal classifier is between 0.5 and 1; if the AUC value of a classifier is lower than 0.5, it means that it is not as good as random guess.

The AUC value is defined as the whole area value under the ROC curve (shown in Figure 1). ROC curve can be obtained by confusing TPR and FPR of matrix. With FPR as the horizontal axis and TPR as the vertical axis, we can obtain the corresponding sensitivity and specificity by giving thresholds. Sensitivity means the probability of a major classification determined as a major classification while specificity means the probability of a minor classification determined as a minor. Assuming that we have a large number of adjustable thresholds, we can get a sensitivity-specificity correlation diagram. That is to say, ROC curve is the trajectory of sensitivity and specificity under different thresholds. The closer the inflection point of ROC curve is to the upper left corner, the larger the area under the curve is, indicating that the model has better effect. On the contrary, the closer the inflection point is to the diagonal line from the upper right to the lower left, the smaller the area under the curve is, indicating that the model is less effective.

Table 2: Evaluation index of ML model.

Evaluation index	Formula
Accuracy rate (AR)	(TP + TN)/(TP + TN + FN + FP)
Recall rate (Recall)	TP/(TP + FN)
Or TPR	—
FPR	FP/(FP + FN)
Precision rate	TP/(TP + FP)
F1 score	2PR/(P + R)

Table 3: Confusion matrix.

		Predicted value 1	Predicted value 0	Total
Actual value	1	True positive rate (TP)	False negative rate (FN)	TP + FN
	0	False positive rate (FP)	True negative rate (TN)	FP + TN
Total		TP + FP	FN + TN	TP + FN + FP + TN

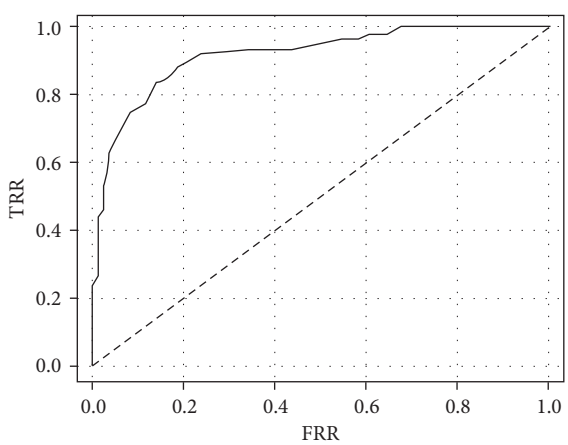


Figure 1: ROC curve graph.

Generally, the AUC value is a probability value to judge whether a model is good or bad. In this paper, we will judge the advantages and disadvantages of the binary classification prediction model with the help of AUC evaluation value. Its evaluation ability is shown in Table 4.

3.3.2. KS Curve. KS curves (shown in Figure 2) are TPR and FPR curves formed under different threshold levels, which are mainly used to verify the distinguishing ability of the model. In the financial risk control, the credit scoring system is constructed, and KS value is often used to measure the performance of the risk control model. Through KS value, we can measure the distinguishing ability of the model from the maximum distance between the cumulative percentage function curve between the correctly predicted borrowers who have not defaulted and the incorrectly predicted borrowers who have overdue. The discrimination ability of KS value is shown in Table 5.

Table 4: AUC evaluation ability classification.

AUC value	Evaluation ability
0.9–1	High accuracy
0.7–0.9	Some accuracy
0.5–0.7	Low accuracy
0–0.5	Not in conformity with the actual situation

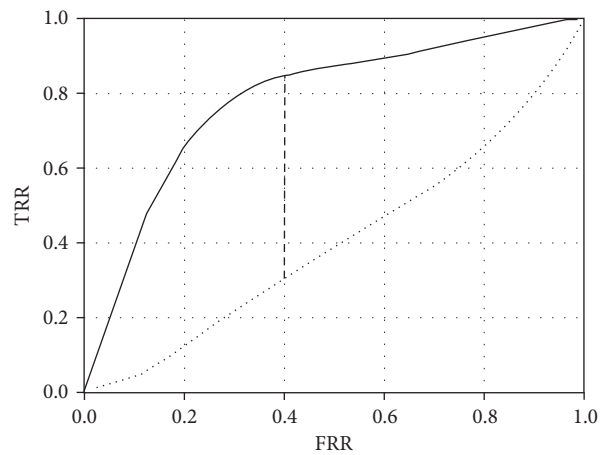


Figure 2: KS curve graph.

Table 5: KS value and model customer identification ability.

KS	The ability to identify risk
<0.2	Null
0.2–0.4	A little
0.41–0.5	Strong
0.51–0.6	Stronger
0.61–0.75	Strongest
>0.75	Abnormal

4. Case Study

In this study, we chose a large P2P Internet lending platform in China as the research case. We analyze the data of 30225 short-term loans from August to December 2018. According to the different performance of the borrowers, they are divided into different categories: D0 represents the borrowers who are not overdue, that is, to repay the principal and interest within the loan term; D1 is overdue less than one month; although they did not repay the loan on time, the overdue time was not too long; if a borrower is overdue for more than one month, D2 is used. For short-term loans, the overdue days are considered as serious overdue because once the customer exceeds these time, the possibility of reperformance is relatively small. The reason for the above classification of borrowers is to better carry out the following analysis and model construction.

In order to facilitate the construction of the evaluation model, a total of 24180 borrowers were classified as D0 and D1 ("good borrowers") and 6045 borrowers were classified

as D2 ("bad borrowers"), which accounts for nearly 80% and 20% of the total sample, respectively. The sample information includes six dimensions including basic information of borrowers such as education, income, age, gender, and so on; credit card transaction records such as billing information and repayment information; debit card payment information; associated loan information; e-commerce platform transaction information; and telecomputer operator information. According to the classification of the six dimensions, we subdivide the variables. However, not all the variable information values meet the minimum threshold set by us, that is, the IV value is greater than 0.02, so we finally selected 372 variables to prepare for modelling.

4.1. Credit Scoring Model

4.1.1. Variable Filtering. For the credit card scoring model, generally only 10–15 variables need to be selected to build the model. Then, you need to filter the variables in advance. The standard for selecting variables is the size of their IV values, and variables with too small IV values are not suitable for selection into the model. In this study, the variables whose IV value is greater than 0.05 and WOE trend is monotonous are selected, and the variables whose correlation coefficient is too high are removed. For example, if the label of the variable is the bank number of the borrower and the IV value is 0.056 through calculation, then this variable will be selected to be added to the credit scoring model. However, the IV value calculated by the average amount of each consumption of the borrower in the last 90 days is 0.026, so the IV value of this variable is too small, and this variable will be eliminated when constructing the model. In addition, the calculated number of transactions consumed in the last 30 days shows that the IV value is 0.082, but the WOE trend is inconsistent, so this variable will also be eliminated.

Following the above ideas, according to IV value, WOE trend, correlation coefficient, and business logic principle, 16 variables are finally selected from different dimensions to consider establishing the model, as shown in Table 6.

4.1.2. Credit Scoring Model. We use the method of logistic regression to build the model because it is easy to monitor and deploy, which is a common method to build credit scoring model. Firstly, we check the coefficients of each variable in the logistic regression method, and it is valid only when the coefficients of variables are positive and variables with negative coefficients will be deleted. Secondly, we set the threshold of p value as 0.05, and if the p value of the variable is greater than this significance level, it will be deleted. Finally, using the programming software, we get the credit scoring model and convert the test set samples, and then we get the scores, as shown in Table 7.

In Table 7, the score represents a range of points. GS is the abbreviation for the number of good samples, followed by the rate represents the ratio of good samples. BS is the abbreviation for the number of bad samples, followed by the ratio of bad samples. TS represents the number of all samples and TR represents the ratio of all samples, and BR refers that bad debt rate.

We can see from Table 7 that with the increase of credit score, the number of good samples in the overall sample shows an upward trend, except for slight decrease in individual intervals, while the number of bad samples is decreasing in general. It shows that good samples should get higher credit scores, while bad samples have lower credit scores. In addition, good samples correspond to lower bad debt rate, while bad samples have higher bad debt rate. From the credit score results, the traditional model has some functions in the credit risk of borrowers.

4.1.3. Model Performance Evaluation. The evaluation of credit scoring model is mainly reflected by KS, AUC, GINI, and other indicators. KS evaluates the model's ability to distinguish customers by calculating the maximum difference between the cumulative percentage of bad customers and good customers; AUC is the standard for judging the advantages and disadvantages of classifiers; GINI coefficient is used to evaluate the risk differentiation ability of the model (Table 8).

As can be seen from Table 8, the KS scores of different types of datasets in the credit scoring model are between 0.3 and 0.4, indicating that the ability of the model to identify customers is not satisfactory and AUC value is between 0.7 and 0.8, indicating that the classifier is better than random guess, and if the model threshold is set properly, there is a certain predictive value; when the GINI value is about 0.5, it indicates that the risk differentiation ability of the model is acceptable.

4.2. ML Model. In this section, the methods related to this work are presented in the following four aspects: data cleaning and feature selection, processing of imbalanced dataset, ML algorithm model setting, and analysis of the result yielded by the proposed ML model.

Step 1. Data cleaning and feature selection.

In data cleaning, we focus on two issues: the processing of empty points and the arrangement of outliers. There are usually four methods to deal with empty points: case deletion method, missing data calculation method, machine learning method, and model-based process [42]. In this study, we mainly deal with empty values based on experience. Specifically, we will delete the features that more than 95% of borrowers did not fill in. At the same time, we will add new features to describe the remaining features. If it is empty, use "1″"; otherwise, use "0." In addition, the average values of these features are calculated to fill the empty points. As far as outliers are concerned, it has been proved that using filters on outliers can improve model performance [43]. By referring to other studies [8], we detect outliers manually and keep reasonable outliers. At the same time, the upper and lower values of the box chart are used to replace the abnormal values. In addition, the feature values are standardized and scaled so that they fall within the specified range of [0, 1].

Table 6: Variables selected into the model.

Bank of deposit	0.053
Number of credit cards	0.052
Maximum credit card limit in recent 1 month	0.084
Maximum overdue days of short-term loans	0.286
The salary per month	0.249
The standard deviation of the number of SMS messages sent at night in the last three months	0.072
The standard deviation of the frequency of answering unlabeled numbers at night in recent two months	0.075
Debit card ratio	0.073
Bill number	0.069
Amount to be paid under credit products	0.065
Average consumption in recent 30 days	0.063
Total data months	0.068
The proportion of credit cards with bills in the last 60 days	0.066
Balance of credit products	0.062
Percentage standard deviation of dialing all numbers at night in recent 60 days	0.060
Bank of deposit	0.061

Table 7: Test set sample score.

Score:	GS	GS rate (%)	BS	BS rate (%)	TS	TR (%)	BR (%)
[low, 575]	366	6.01	542	24.63	908	10.02	51.28
[576, 585]	518	8.31	396	15.32	914	9.65	32.32
[586, 595]	722	9.35	345	15.2	1067	10.86	30.19
[595, 601]	699	9.35	263	10.36	962	9.52	21.25
[602, 608]	885	10.52	132	9.52	1017	10.27	18.77
[609, 615]	887	10.36	106	7.02	988	9.65	14.38
[616, 622]	896	10.89	96	5.99	992	9.68	12.35
[623, 630]	902	11.96	90	5.26	992	10.05	10.37
[631, 641]	932	11.68	68	3.88	1000	10.01	7.85
[642, high]	941	12.05	35	1.38	976	9.68	2.73

Table 8: Result evaluation of credit scoring model.

Data classification	KS	AUC	GINI
Training set	0.3628	0.7269	0.4696
Validation set	0.3265	0.7225	0.4426
Testing set	0.3269	0.7244	0.4535

The next work is feature selection which could improve the operation efficiency and the prediction result of classifier. Generally, subset selection can be used to improve the performance of feature selection process, such as wrappers, filters, and embedding method [44, 45]. According to reference [8], this study adopts a tree-based feature selection method, which is an embedded method, namely, feature selection based on random forest model [46]. Random forest can be used not only to calculate the importance of different features but also to delete irrelevant features.

Step 2. Processing of imbalanced data.

Most of the studies on credit risk assessment models for Internet financial institutions are based on imbalanced data, which means the number of nondefault cases is usually larger than the default ones; if we ignore the class imbalance problem to buid a classification model, we might obtain a model that has high accuracy for the determination of nondefaults but extremely low accuracy for default. To solve this issue, this paper tries to deal with SMOTE algorithm.

In the dataset of this study, because the sample size of defaulting borrowers accounts for less than 10% of the total sample, it belongs to unbalanced sample. If the misjudgment rate is used as the evaluation index of the model, the data in this paper may have a relatively large risk, and it is impossible to get a valuable model. The SMOTE algorithm artificially synthesizes new samples based on a small number of samples, and adds the synthesized new samples to the data set. The basic idea of SMOTE algorithm is to find the distribution space of small class samples according to the partial characteristics of two kinds of samples in p-dimensional space and finally generate new small class samples between small class samples and small class samples. Referring to [47], the algorithm flow is as follows:

(i) Taking Euclidean distance as the standard, for each sample $\$ \times \$$ in a small sample class, the distance from it to all samples in the minority sample set S_min is calculated, and its k-nearest neighbour is obtained.

(ii) The sampling rate is set according to the sample imbalance ratio to determine the sampling rate N.

(iii) For each minority sample $\times$, it is randomly selected from its k-nearest neighbours, if the selected nearest neighbour is $/hat{$x$}$.

(iv) For every randomly selected nearest neighbour $/hat{x}$, build new sample by the formula

$$x_{\text{new}} = x + r \text{ and } (0, 1) \times (\tilde{x} - x). \tag{15}$$

The specific idea of the algorithm is shown in Figure 3.

By using SMOTE algorithm, the data distribution in this paper tended to be balanced. The ratio of expected default to expected nondefault is 1 : 1.33, which makes the sample category basically balanced.

Step 3. ML model setting.

In this section, we employ grid search to set a series of hyperparameters, which is a fundamental parameter optimization method. And it will substantially divide the hyperparameter into the grids with same length in the certain rang of coordinate system. Every point in the coordinate system represents a set of hyperparameters, and then we could adopt every point in a certain interval into our model to verify the performance of the algorithm. The point that performs best is called best hyperparameter. In other word, the algorithm of grid search is to traverse the points corresponding to all grids.

Grid search was used to optimize the combination of hyperparameters within 5 cross-validations [47]. Since grid search uses an exhaustive search of predefined hyperparameter space, we provide the search space for these algorithms here: number of iterations was set in the range of 100 to 500, the depth of trees is in the range of 5 to 25, and learning rate is in the mathematical set of (0.001,0.01,0.1,1). The lowest gradient descent of loss function is set to 0. Besides this, SONNIA (2016) was used to generate the SOMs in this work [48]. Parameters were set as default.

Step 4. Results and analysis.

The data processing speed of machine learning is calculated by programming: "start = time. Perf _ counter ( ), End = time. Perf_counter ( ), T = end-start," and we get the speed of the ML model to deal with selected variables is 9 milliseconds which is very fast.

The application of ML model should consider the actual business situation of the organization. For Internet financial institutions, setting up a strict preloan approval system and granting loans only to customers with high credit scores can reduce the credit risk to some extent, but it will lead to a large number of customers unable to carry out transactions due to lack of qualifications, which will affect their business results. Based on this situation, this experiment considers different prediction results of ML model under different preset probabilities, as shown in Table 9.

In Table 9, PP is an abbreviation for the preset probability value. GS represents cumulative good samples, and GB represents cumulative bad samples. The passing rate is

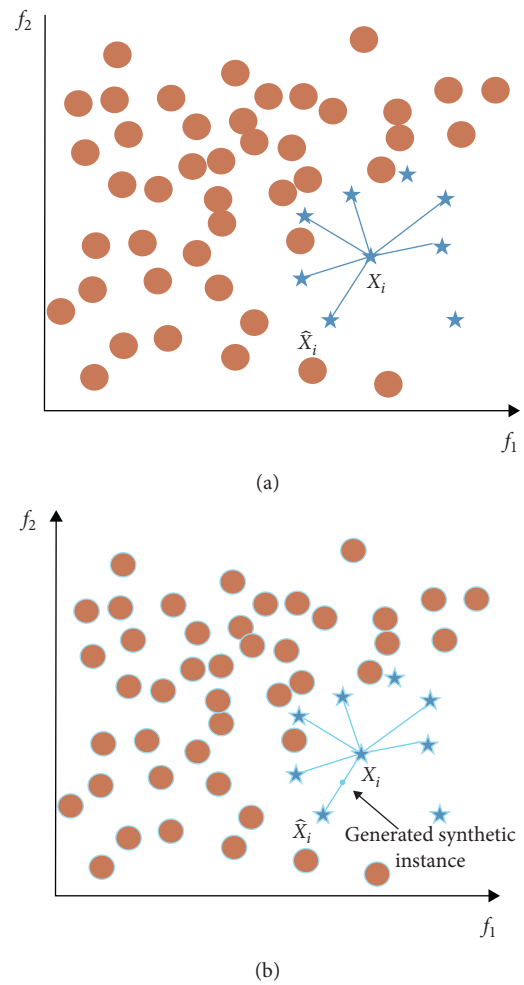


Figure 3: SMOTE algorithm principle.

expressed by PR, and ER stands for the error rate. The values of each item are reserved with four digits after the decimal point. It can be seen from Table 8 that the model can achieve the highest KS value of 0.4936 with the preset probability of 0.6~0.65, which means that a loan customer can pass the screening only when the probability of predicting a good customer is greater than 0.6. Under this standard, 59.98% of the applicants have passed the loan application. However, the error rate of the model is 7.44%, which means that 7.44% of the bad samples are wrongly judged as good samples.

Because the low pass rate will also affect the financial performance, the operators engaged in the Internet finance industry should comprehensively measure and compare the KS value, pass rate, and misplacement rate from the perspective of realizing business and then choose a preset probability threshold that best meets its operating conditions. By comparing Tables 8 and 9, we can see the difference between the traditional credit scoring model and the ML model. Under the preset probability of 0.6, the KS value of traditional credit scoring model is 0.3269, while the result of ML model is 0.4936. This shows that under this preset probability, the prediction ability of ML model is obviously better than that of traditional credit card scoring model.

Table 9: ML model prediction results.

PP	GS	DS	KS	PR	ER
[0.50, 0.55]	0.1444	0.5269	0.4165	0.7023	0.0992
[0.55. 0.60]	0.1818	0.5986	0.4930	0.6032	0.0861
[0.60, 0.65]	0.2635	0.6320	0.4936	0.5998	0.0744
[0.65, 0.70]	0.2895	0.6598	0.4360	0.5366	0.0634
[0.70, 0.75]	0.3265	0.7998	0.3963	0.5183	0.0588
[0.75, 0.80]	0.3984	0.7911	0.3641	0.4880	0.0481
[0.80, 0.85]	0.4698	0.8698	0.3201	0.4609	0.0307
[0.85, 0.90]	0.5024	0.9269	0.3004	0.3504	0.0287

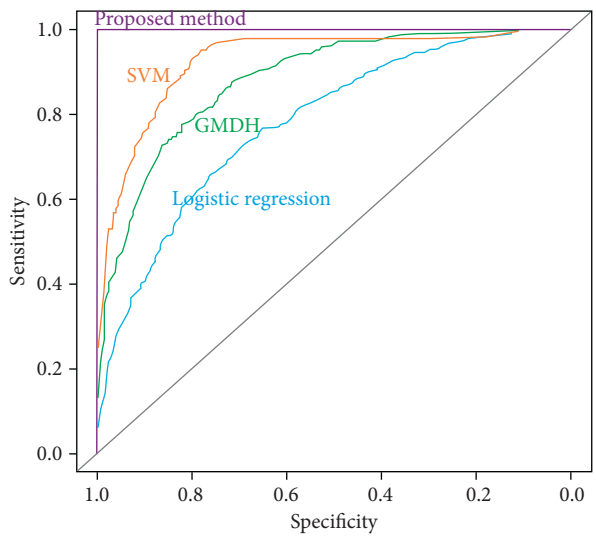


Figure 4: ROC from same training data on different classifiers.

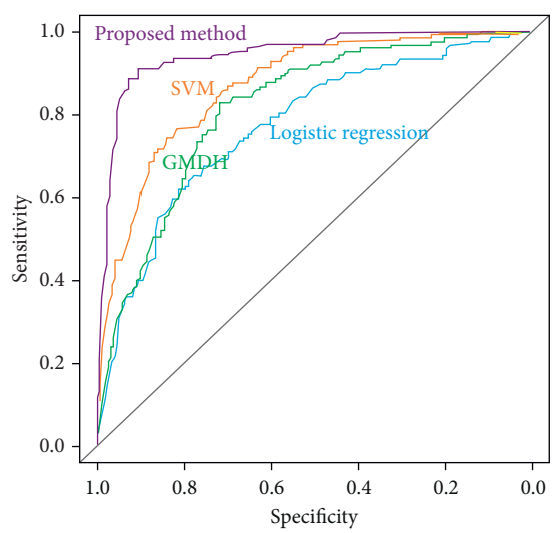


Figure 5: ROC from same test data on different classifiers.

4.3. Validation of the XGBoost ML-Based Model in This Paper. In order to verify the effectiveness of ML algorithm in the credit risk management of Internet financial industry, this paper selects a large Internet financial lending platform in China as a research case and compares the performance of the traditional model and the model proposed in this paper. In order to further enhance the comparability of the model, more methods are introduced to compare the simulation and experimental results. We compare the results of credit scoring model which based on logistic regression, neural network method and support vector machine learning method for data grouping processing [49] with the results of the method proposed in this study [50]. Referring to other research ideas [30], we compare the simulation figures of

Table 10: Comparison of accuracy rates yielded by different classifiers.

Model	Cutoff point	Training data			Test data		
		Classification	Discriminant accuracy	Total accuracy	Classification	Discriminant accuracy (%)	Total accuracy (%)
Logistic regression	0.35	Nondefaults	72.9	71.5	Nondefaults	69.7	70.1
		Defaults	70.1		Defaults	70.5	
GMDH	0.41	Nondefaults	85.1	79.4	Nondefaults	75.1	75.1
		Defaults	73.7		Defaults	75.1	
SVM	0.26	Nondefaults	88.3	83.1	Nondefaults	78	77.4
		Defaults	77.9		Defaults	76.7	
Proposed method	0.6	Nondefaults	100	100	Nondefaults	89.6	90.1
		Defaults	100		Defaults	90.6	

different experimental results. Figures 4 and 5 show the ROC based on training set and test set data, respectively, and Table 10 shows the classification accuracy, which comes from the optimal cutoff point 0.6 when default accuracy equates to nondefault accuracy based on test data.

We can see from Figures 4 and 5 that the AUC value of XGBoost classifier is the best based on the same test data. At the same time, Table 10 shows that the overall accuracy rate of the proposed Internet financial risk assessment model is the best (90.1%), which is better than the traditional logistic regression model (70.1%), support vector machine (77.4%), and GMDH (75.1%). When dealing with the same dataset and training set, the performance of this method is better than other classifiers.

5. Conclusions

In this paper, we propose an improved ML-based technique for credit card scoring in Internet financial risk control, which has better performance than the traditional credit scoring modern in Internet financial risk control. Because the traditional credit evaluation model is complicated and has strict research on the selection of variables, it has some limitations. And this method has strict data requirements that in the Internet age, there is a limitation that it cannot analyze the personal credit data with high dimension, high complexity, and nonlinearity. However, with the deep integration of Internet era and traditional financial industry, the vigorous development of Internet financial industry is the inevitable trend of social development. At the same time, financial institutions engaged in Internet financial business will process a large number of customer data, which is more important for the control of credit risk. Therefore, we must consider which method to use to carry out the credit risk of the Internet financial industry, and ML algorithm has become a good alternative. The main contribution of this paper is to propose the application of ML algorithm to financial risk control in the field of Internet finance because it can show better performance than the traditional credit scoring model and better match with the background of big data. Therefore, this paper has a certain reference value for the risk management practice of the Internet financial industry.

The proposed ML model is tested on an Internet financial platform in China. The experimental results indicated that the process of building up model and dealing with data is more efficient. Compared with the traditional credit scoring model, ML algorithm can process a large number of data in a very short time to meet the requirements of Internet financial institutions to process a large number of customer information. In addition, there are no strict restrictions on the data processed by ML algorithms. In order to improve the performance of the model prediction results, we can set model parameters in advance, add variables to the model, and then eliminate the variables that contribute less to the model according to the importance of features. The experimental results show that only when the probability that a loan applicant is predicted to be a good customer is greater than 0.6, can the loan application be screened. At this time, the KS value obtained by the ML model is 0.4936, which exceeds the KS value of the traditional credit scoring model of 0.3269. This indicates that the ML model has certain advantages in the application of Internet financial risk control.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

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

References

[1] S. Yu, B. Bing, H. Peikai et al., "The study of the tourism enterprises' financing Capacity under the background of internet, travel and finance commune," *Finance Account*, vol. 11, no. 24, pp. 3-4, 2017.

[2] J. Zhang and Q. Sun, "Research on financing cost of small and medium-sized enterprises by internet finance Open," *Open Journal of Social Sciences*, vol. 48, no. 25, p. 95, 2017.

[3] Y. Lin and C. Chen, "Research on enterprise financial risk evaluation based on association rules," *Friends Account*, vol. 96, no. 2365, pp. 32–35, 2017.

[4] J. LiuJ.A. Xiuyi et al., "Multi-label classification algorithm based on association rules mining," *Journal of Software*, vol. 28, no. 11, pp. 2865–2878, 2017.

[5] Nonymousm and Liu, "Research on bank product recommendation model based on big data mining in fintech era China," *Finance Computer*, vol. 233, no. 4356, pp. 38–40, 2018.
[6] J. Zhao, "Internet Finance and its risk prevention and control," *Tax and Economy*, vol. 23, no. 2336, pp. 52–63, 2018.
[7] L. C. Thomas, J. Crook, D. Edelman et al., "Credit scoring and its applications," *Society for Industrial and Applied Mathematics*, vol. 579, no. 3487, pp. 221–225, 2012.
[8] B. Wang, L. Ning, Y. Kong et al., "Integration of unsupervised and supervised machine learning algorithms for credit risk assessment," *Expert Systems with Applications*, vol. 128, pp. 301–315, 2019.
[9] S. Firdous and R. Farooqi, "Impact of internet banking service quality on customer satisfaction," *Journal of Internet Banking and Commerce*, vol. 121, no. 2201, pp. 1–17, 2017.
[10] H. R. Khedmatgozar and A. Shahnazi, "The role of dimensions of perceived risk in adoption of corporate internet banking by customers in Iran Electron," *Electronic Commerce Research*, vol. 2, no. 18, pp. 389–412, 2018.
[11] H. A. AbdouGenetic, "programming for credit scoring: the case of Egyptian public sector banks," *Expert Systems with Applications*, vol. 9, no. 36, pp. 11402–11417, 2009.
[12] D. J. Hand and W. E. Henley, "Statistical classification methods in consumer credit scoring: a review," *Journal of the Royal Statistical Society: Series A*, vol. 3, no. 160, pp. 523–541, 1997.
[13] S. Lee, C.-C. Chiu, Y.-C. Chou, and C.-J. Lu, "Mining the customer credit using classification and regression tree and multivariate adaptive regression splines," *Computational Statistics and Data Analysis*, vol. 163, no. 2569, pp. 1113–1130, 2006.
[14] W. Y. Bee, H. O. Seng, N. Huselina Mohamed Husain et al., "Using data mining to improve assessment of credit worthiness via credit scoring models," *Expert Systems with Applications*, vol. 38, no. 10, pp. 13274–13283, 2011.
[15] G. Williams and Z. Huang, "Mining the knowledge mine: the hot spots methodology for mining large real world databases," in *Proceedings of the 10th Australian Joint Conference on Artificial Intelligence*, Perth, Australia, November 1997.
[16] A. Yeo, K. Smith, R. Willis, M. Brooks et al., "Clustering technique for risk classification and prediction of claim costs in the automobile insurance industry," *Intelligent Systems in Accounting Finance*, vol. 322, no. 2986, pp. 39–50, 2001.
[17] E. Khandani, A. J. Kim, A. W. Lo et al., "Consumer credit-risk models via machine-learning algorithms," *Journal of Banking & Finance*, vol. 344, no. 4356, pp. 2767–2787, 2016.
[18] C. Chakraborty and A. Joseph, "ML at central banks," *Bank of England Staff Working*, vol. 674, 2017.
[19] J. Ticknor, "Bayesian regularized artificial neural network for stock market forecasting," *Expert Systems with Applications*, vol. 14, no. 40, pp. 5501–5506, 2013.
[20] T. P. Gogas and A. Agrapetidou, "Forecasting bank failures and stress testing: a ML approach," *International Journal of Forecasting*, vol. 34, no. 13, pp. 440–455, 2018.
[21] N. Rtayli and N. Enneya, "Selection features and support vector machine for credit card risk identification," *Procedia Manufacturing*, vol. 46, pp. 941–948, 2020.
[22] P. Pławiak, M. Abdar, J. Pławiak et al., "DGHNL: A new deep genetic hierarchical network of learners for prediction of credit scoring," *Information Sciences*, vol. 516, pp. 401–418, 2020.
[23] H. Hou and S. Liu, "Credit risk assessment of commercial banks based on support vector machine," *Computer Engineering and Application*, vol. 31, no. 40, pp. 176–178, 2004.
[24] J. Hou, Q. Xue, P. Xu et al., "Prediction of credit risk of personal housing loan by approximate support vector machine," *Journal of Xi'an University of Technology*, vol. 6, no. 31, pp. 559–565, 2011.
[25] H. Hu, L. Zhang, D. Zhang et al., "Research on credit risk assessment of supply chain finance based on support vector machine," *Soft Science*, vol. 5, no. 25, pp. 26–30, 2011.
[26] T. Zhao and W. Chen, "Credit risk re assessment of retail business of commercial banks based on credit consumption behavior," *Financial Theory and Practice*, vol. 12, no. 37, pp. 75–79, 2016.
[27] X. Liu, J. Tang, Z. Duan et al., "Research on feature selection of AUCRF algorithm in credit risk evaluation," *Computer Applications and Software*, vol. 4, no. 35, pp. 293–295, 2018.
[28] J. L. Bellovary, D. E. Giacomino, M. D. Akers et al., "A review of bankruptcy prediction studies: 1930 to present," *Financial Education*, vol. 42, no. 33, pp. 1–42, 2010.
[29] H. A. Abdou and J. Pointon, "Credit scoring, statistical techniques and evaluation criteria: a review of the literature," *Account and Financial Manage*, vol. 18, no. 3, pp. 59–88, 2011.
[30] Y.-C. Chang, K.-H. Chang, G.-J. Wu et al., "Application of eXtreme gradient boosting trees in the construction of credit risk assessment models for financial institutions," *Applied Soft Computing*, vol. 73, pp. 914–920, 2018.
[31] W. Lin and Y. Hu, "ML in financial crisis prediction: a survey," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 4, no. 42, pp. 421–436, 2012.
[32] X. Wang, M. Xu, Ö.T. Pusatli et al., "A survey of applying ML techniques for credit rating: existing models and open issues," *Neural Information Processing*, vol. 98, no. 56, pp. 122–132, 2015.
[33] F. Louzada, A. Ara, G. B. Fernandes et al., "Classification methods applied to credit scoring:Systematic review and overall comparison," *Surveys in Operations Research and Management Science*, vol. 2, no. 21, pp. 117–134, 2016.
[34] S. Devi and Y. Radhika, "A survey on ml and statistical techniques in bankruptcy prediction," *Knowledge-Based Systems*, vol. 22, no. 67, pp. 120–127, 2018.
[35] M. BrunoHenrique et al., "Literature review: ML techniques applied to financial market prediction," *Expert Systems with Applications*, vol. 124, pp. 226–251, 2019.
[36] Credit Risk Scorecards, *Developing and Implementing Intelligent Credit Scoring*, SAS Publishing, Cary, NC, USA, 2005.
[37] I. J. Myung, "Tutorial on maximum likelihood estimation," *Mathematical Psychology*, vol. 1, no. 47, pp. 90–100, 2003.
[38] F. A. Qi de, Xu L. Cheng, Z. Zhu et al., "Xgboost recommendation algorithm based on collaborative filtering," *Computer Application Research*, vol. 5, no. 37, pp. 1317–1320, 2020.
[39] L. Song, S. Wang, C. Yang et al., "Application of improved XGBoost in unbalanced data processing," *Computer Science*, vol. 6, no. 47, pp. 98–103, 2020.
[40] Liu and Chen, "Loan risk prediction method based on SMOTE and XGBoost," *Computer and Modernization*, vol. 2, pp. 26–30, 2020.
[41] Y. Cui, W. Qi, H. Pang et al., "Recommendation algorithm combining collaborative filtering and xgboost," *Computer Application Research*, vol. 1, no. 12, pp. 62–65, 2020.
[42] P. J. García-Laencina, J. L. Sancho-Gómez, A. R. Figueiras-Vidal et al., "Pattern classification with missing data: a review," *Neural Computing and Applications*, vol. 2, no. 19, pp. 263–282, 2010.
[43] V. García, A. I. Marqués, J. S. Sánchez et al., "On the use of data filtering tech- niques for credit risk prediction with

instance-based models," *Expert Systems with Applications*, vol. 39, no. 18, pp. 13267–13276.

[44] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2011.

[45] Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, vol. 19, no. 23, pp. 2507–2517, 2007.

[46] L. Breiman, "Random forest," *Machine Learning*, vol. 46, no. 327, pp. 1–35, 1999.

[47] X. Pan, "Prediction and analysis of credit risk of P2P online loan borrowers," Guizhou University of Finance and Economics, Guiyang, China, 2019pp. 34-35, Master thesis.

[48] Sonnia, *Molecular Networks GmbH: Germany and Altamira*, vol. 2, LLC, New York, NY, USA, 2016, https://www.mn-am.com/products/sonnia.

[49] A. G. Ivakhnenko, "The group method of data handling–a rival of the method of stochastic approximation," *Soviet Automatic Control*, vol. 13, pp. 43–55, 1996.

[50] B. E. Boser, I. M. Guyon, V. N. Vapnik et al., "A training algorithm for optimal margin classifiers," in *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pp. 144–152, ACM Press, New York, NY, USA, July 1992.

Hindawi
Complexity
Volume 2020, Article ID 7141320, 13 pages
https://doi.org/10.1155/2020/7141320

Research Article

A Methodology for Calculating Greenhouse Effect of Aircraft Cruise Using Genetic Algorithm-Optimized Wavelet Neural Network

Yong Tian,[1] **Lina Ma**,[1] **Songtao Yang**,[2] **and Qian Wang**[1]

[1]*College of Civil Aviation, Nanjing University of Aeronautics and Astronautics, Nanjing 211100, China*
[2]*School of Mechanical Engineering, University of Leeds, Leeds, LS2 9JT, UK*

Correspondence should be addressed to Lina Ma; malinacca@nuaa.edu.cn

Received 21 July 2020; Revised 2 September 2020; Accepted 8 October 2020; Published 29 October 2020

Academic Editor: Min Xia

Reliable assessment on the environmental impact of aircraft operation is vital for the performance evaluation and sustainable development of civil aviation. A new methodology for calculating the greenhouse effect of aircraft cruise is proposed in this paper. With respect to both cruise strategies and wind factors, a genetic algorithm-optimized wavelet neural network topology is designed to model the fuel flow-rate and developed using the real flight records data. Validation tests demonstrate that the proposed model with preferred network architecture can outperform others investigated in this paper in terms of accuracy and stability. Numerical examples are illustrated using 9 flights from Beijing Capital International Airport to Shanghai Hongqiao International Airport operated by Boeing 737–800 aircraft on October 2, 2019, and the generated fuel consumption, CO_2 and NO_x emissions as well as temperature change for different time horizons can be effectively given through the proposed methodology, which helps in the environmental performance evaluation and future trajectory planning for aircraft cruise.

1. Introduction

As a typical manifestation of the negative externalities of air transport, environmental impact is easily overlooked in management decisions, while it greatly affects the sustainability of operation. With the prosperity of global air transport industry, the dramatically increasing flights have resulted in prominent environmental issues [1]. Statistically, aviation is responsible for 13% of fossil fuel consumption related to transportation, 2% of the anthropogenic carbon dioxide emissions [2], and 24% of global nitrogen oxide emissions [3], and the climatic change caused by it will rise 3%–11% in the next 30 years [4]. Meanwhile, a further increase of these figures can be foreseen for the expected growth of global air traffic at an annual rate of 4.8% from 2011 to 2030 [5]. Under the determination in promoting green civil aviation development of the international community, it is imperative to focus on and study about the environmental impact caused by aircraft operation.

In order to address and evaluate the aviation environmental impact, current studies are conducted mostly from the perspective of gas emissions. The baseline emission model provided by the International Civil Aviation Organization (ICAO) was applied to estimate the landing and take-off (LTO) emissions of civil airports in China by Xia et al. [6] and to predict the future CO_2 and NO_x emissions of the air transportation by Owen et al. [7]. Wei et al. [8] calculated pollutant emissions of aircraft in different cruise conditions using Boeing Performance Software (BPS). Torija and Self [9] proposed an Environmental Impact Aviation Metric (EIAM) method to calculate the LTO emissions. Recently, the 2018–2019 inventory of air pollutant emissions for the Beijing-Tianjin-Hebei Airport group in China was obtained by Han et al. [10] using operational data. These studies have calculated the aircraft gas emissions from different spatial and temporal dimensions, while little research has been further conducted in terms of the temperature change caused by them. Essentially, the

environmental impact of gas exhausted is the greenhouse effect that contributes to global warming, and the issue of global warming has attracted public extensive attention gradually [11]. In addition, it should be noted that cruise phase should attract more attention because the time spent on it accounts for the vast majority of the total, and the greenhouse effect derived from exhaust emissions is amplified at a high altitude [12].

As is widely acknowledged, aircraft exhaust emissions contributing to greenhouse effect derive from the fuel combustion in aero-engines, meaning that the acquisition of fuel consumption value is prominent in environmental impact assessment. Numerous researches on fuel consumption modeling have been carried out for its largest portion of operational costs or significant detrimental impact to environment. The methods used are mainly classified into two categories consisting of mathematical programming and machine learning. For mathematical programming methods, user manual for the Base of Aircraft Data (BADA) [13] of Eurocontrol presents a method that calculates the nominal FFR using the thrust and thrust-specific fuel consumption, which functioned by the true airspeed. And it was utilized to model the cruise fuel consumption and improved through determining new empirical constants with regard to the Mach number by Bartel and Young [14]. In the research of Senzig et al. [15], a model of fuel consumption in terminal area was proposed and verified with the consideration of temperature ratio, Mach number, and net corrected thrust. Turgut and Rosen [16] investigated an exponential relationship between fuel flow rate (FFR) and altitude for the descent phase. And there were also a series of regression models for fuel consumption estimation based on the Gaussian and K-nearest neighbour (KNN) methods as presented by Lawrance [17]. From the studies above, the mathematical programming models are mostly based on the engine thrust and rely on a large database of aircraft performance and flight parameters with significant complexity in calculating and limited accuracy of results.

In recent decades, another data-driven method of machine learning also shows efficiency in FFR modeling. Backpropagation (BP) neural networks based on flight parameters for FFR prediction in different flight phases were presented by Liu [18]. Although the application of neural network significantly simplified the estimation process compared with mathematical programming, there were a series of defects such as its slow convergence speed and easiness to fall into local optimum. Later, Zhang and Xu [19] and Baklacioglu et al. [20] optimized the classical BP neural networks using particle swarm and genetic algorithm, respectively, allowing higher accuracy available, but the requirement for detailed data about aircraft operation and associated state greatly limited the practicality of the model. For example, the data of aircraft pitch angle, roll angle, and true airspeed set as input parameters in their models are always difficult to obtain in application. In this regard, trajectories parameters were employed to construct a BP neural network for FFR by Wei and Zhang [21], through the network the aircraft FFR was output given its ADS-B tracking information which was much more accessible. Zhou [22] explored the relationship between flight altitude, ground speed, and true airspeed and developed a true airspeed estimation model, based on which a FFR model was further constructed using the deep belief network (DBN). It should be noted that the meteorological factors playing important parts in fuel consumption were excluded in their models.

In summary, there are still deficiencies in the metrics and accuracy of aviation environmental impact assessment. In order to achieve the concept of "green flight", it is practical and far-reaching to develop an effective methodology of greenhouse effect calculation for aircraft cruise, for which a calculation method based on genetic algorithm-optimized wavelet neural network (GA-WNN) is proposed in this paper. This paper contributes to aviation environmental impact evaluation from three aspects. Firstly, a supervised learning method based model considering the meteorological factors is constructed for aircraft FFR prediction, contributing to reduced dependency on detailed operation data and more conformance with the reality. Secondly, the genetic algorithm is initiatively used to optimize the architecture of wavelet neural network applied to FFR modeling, which leads to a further improvement for network performance. On top of the above improvements, a feasible and reliable methodology focused on temperature change metric for environmental impact assessment is formed, providing supports for developing sustainable civil aviation.

The reminder of this paper is organized as follows. Section 2 formulates the calculation of the greenhouse effect caused by aircraft cruise to a linear function. In Section 3, a genetic algorithm-optimized wavelet neural network model is proposed, and the developing procedure for it using real raw flight records data of the aircraft type, and Boeing 737–800 is presented in detail. In Section 4, applications of the developed model to 9 flights are represented as numerical examples, followed by the last section, which summarizes the research with concluding remarks and gives an outlook to the future work.

2. Problem Formulation

For the environmental impact quantification, CO_2 and NO_x are considered in this paper because a vast majority of the greenhouse effect that aircraft causes when cruising comes from them. Based on their emissions and using the global absolute temperature change potential (AGTP) as a parameter, a linear function formulating the caused greenhouse effect is developed in this section.

2.1. Greenhouse Gas Emission

2.1.1. CO_2 Emissions. According to the Emissions and Dispersion Modeling System (EDMS) published by Federal Aviation Administration (FAA), the CO_2 emissions can be functioned by its emission index and FFR as follows:

$$E_{CO_2} = \mathrm{EI}_{CO_2} \int_0^{T_c} \mathrm{FFR}(t)\mathrm{d}t, \tag{1}$$

where E_{CO_2} is the value of CO_2 emissions, EI_{CO_2} is the emission index of CO_2 and is only relevant to the engine type, T_c is the total time (h) spent in cruise phase, and $FFR(t)$ (kg/h) is the value of FFR at the moment of t.

2.1.2. NO_x Emissions. The NO_x emissions during cruise phase is also closely related to FFR and NO_x emission index EI_{NO_x}, but the difference from CO_2 emissions is that the value of EI_{NO_x} depends additionally on the atmospheric conditions and FFR besides the engine type. Therefore, EI_{NO_x} changes with the moment t during the cruise phase.

In order to obtain the real-time EI_{NO_x}, a method combining interpolation using the specific FFR with modification according to atmospheric conditions is proposed in this paper, and that includes the following three steps.

Step 1: convert the FFR value in actual operational conditions to that in standard conditions (ISA, 0m) according to

$$FFR' = \frac{FFR}{\delta}\theta^{3.8}e^{0.2}M^2, \tag{2}$$

where FFR' is the converted value of FFR, δ is the ratio of local atmospheric pressure to that on the standard sea level (1013.2 hPa), θ is the ratio of local atmospheric temperature to that on the standard sea level (288K), e is the natural constant valuing about 2.72, and M is the Mach number [8].

Step 2: get a fitting relationship between FFR and the NO_x emission index in standard conditions with the use of basic emission data of engine so that the baseline NO_x emission index EI'_{NO_x} can be valued from FFR' through the relationship. For example, the fitting formula for CFM56-7B26 installed on Boeing 737–800 aircraft is as follows when FFR' values are between 300 and 5 000:

$$EI'_{NO_x} = p_1 + p_2 \cdot FFR' + p_3(FFR')^3 + p_4 \ln(FFR'), \tag{3}$$

where p_1, p_2, p_3, and p_4 are fitting coefficients valuing -0.0283, -2.1745×10^{-7}, -1.3976×10^{-13}, and 0.0055, respectively. They are obtained by the universal global optimization (UGO) algorithm and contribute to a R^2 up to 0.9999 for (3).

Steps 3: modify EI'_{NO_x} back to the value in actual operating conditions using the following equation:

$$EI_{NO_x} = EI'_{NO_x}\frac{\delta^{0.51}}{\theta^{1.65}}\exp\left[19.0\left(0.0063 - \frac{0.622\varphi p_v}{p - \varphi p_v}\right)\right], \tag{4}$$

where φ (%) is the atmospheric relative humidity, p_v (Pa) is the saturated vapor pressure, and it can be calculated through the atmospheric temperature $T(K)$ as follows:

$$\lg p_v = \frac{10286T - 2148.4909}{T - 35.85}. \tag{5}$$

Once the emission indices at each moment $EI_{NO_x}(t)$ are obtained, combined with the fuel flow rates $FFR(t)$, the NO_x emissions E_{NO_x} can be derived as follows:

$$E_{NO_x} = \int_0^{T_c} EI_{NO_x}(t) \cdot FFR(t)dt. \tag{6}$$

2.2. Greenhouse Effect Characterization. There are several climate indicators to assess the impact of gas emissions. The absolute global temperature potential (AGTP), denoting the mean change of surface temperature caused by per unit greenhouse gases, is used in this paper to uniformly characterize the environmental impact of various gases including CO_2 and NO_x, which can result in different temperature increases for their difference in radiative forcing and life cycles. The AGTP from gas G for the time horizon of H years is expressed as $AGTP_G(H)$, and it can be calculated through

$$AGTP_{G(H)} = \begin{cases} A_G \sum_{j=1}^{2}\left[a_0 c_j\left(1 - e^{-(H/d_j)}\right) + \sum_{i=1}^{3}\frac{a_i\alpha_i c_j}{\alpha_i - d_j}\left(e^{-(H/\alpha)} - e^{-(H/d_j)}\right)\right], & G = CO_2, \\ A_G \sum_{j=1}^{2}\left[\frac{\alpha c_j}{\alpha - d_j}\left(e^{-(H/\alpha)} - e^{-(H/d_j)}\right)\right], & G = NO_x, \end{cases} \tag{7}$$

where G represents the exhausted gas, namely, CO_2 or NO_x, A_G (Wm^2/kg) is the radiative forcing change per unit G causes, and a_0, c_j, d_j, a_i, α_i, and α are all given parameters [23].

Then the surface temperature change $T(H)$ due to CO_2 and NO_x emissions during cruise phase can be formulated as a linear function:

$$\begin{aligned} \Delta T(H) &= AGTP_{CO_2}(H) \cdot E_{CO_2} + AGTP_{NO_x}(H) \cdot E_{NO_x}, \\ &= AGTP_{CO_2}(H) \cdot EI_{CO_2}\int_0^{T_C} FFR(t)dt \\ &\quad + AGTP_{NO_x}(H)\int_0^{T_C} EI_{NO_x}(t) \cdot FFR(t)dt. \end{aligned} \tag{8}$$

3. GA-WNN Model for Fuel Flow-Rate

As the linear function developed previously, the aircraft FFR at each moment during cruise phase is the most critical factor for the greenhouse effect calculation. In this section, a GA-WNN model for FFR based on the wavelet neural network (WNN) technology is proposed, followed by a detailed developing procedure using the real flight records data of Boeing 737–800 aircraft.

3.1. Wavelet Neural Network. As discussed by Lai et al. [24], the wavelet neural networks (WNNs) were developed based on the technology of BP neural network and they were similar in topology. However, the original sigmoid function as the transfer function for hidden layer nodes was substituted by the wavelet basis function in WNN, and the shift and scaling factors were introduced to replace the corresponding weights and thresholds respectively. Combined with the time-frequency locality of the wavelet analysis, the WNN has improved capability in self-learning, robustness, and adaptability, resulting in a better prediction performance.

In order to overcome the shortcomings of the previous methods that depend too much on those less publicly available data involved in airline operation details and to further improve the performance of the supervised learning based model, a three-layer WNN was constructed to model FFR of the transport aircraft during cruise phase in this paper. The WNN topology is shown in Figure 1.

In the input layer X, there are five neurons defined based on the aircraft state and meteorological wind factors affecting FFR in cruise phase, including the pressure altitude, the ground velocity, the heading of aircraft, and the speed and direction of the wind. With the normalized input data h, v, d, ws, and wd (see Figure 1), the output value of node y_j in the hidden layer Y, $y(j)$ can be expressed as follows:

$$y(j) = f_j\left(\sum_{i=1}^{5} w_{ij}x_i\right), \quad j = 1, 2, \ldots, J, \tag{9}$$

where J is the total nodes number in Y, w_{ij} is the connection weight between node x_i in X and node y_j in Y, and f_j is the wavelet basis function, derived from the scaling and shift translation of a mother wavelet Ψ_j. Given a scaling factor a_j and a shift factor b_j, the mathematical expression of $y(j)$ can be rewritten as

$$y(j) = \Psi_j\left[\frac{\sum_{i=1}^{5} w_{ij}x_{i-}b_j}{a_j}\right], \quad j = 1, 2, \ldots, J. \tag{10}$$

The neuron in the output layer Z is the very factor that is going to be modeled, namely, the aircraft FFR, and its predicted value z^{pre} output from the network is calculated by

$$z^{\text{pre}} = \sum_{j=1}^{J} w_{jk}y(j), \quad k = 1, \tag{11}$$

where w_{jk} is the linked weight between node y_j in Y and node z in Z and k takes the constant value of 1 for there are only one node in Z as Figure 1 shows.

Similar to BP neural network, the WNN begins working with randomly initializing the connection weights between layers and the factors of wavelet basis function, following which the output value can be calculated forward layer by layer according to (10) and (11). Later the output error e of a single simple can be written as follows:

$$e = |z^{\text{pre}} - z^{\text{exp}}|, \tag{12}$$

where z^{exp} is the expected or actual value of FFR.

Through the backpropagation of e, the initial weights and factors are constantly corrected using a gradient learning method as (13)–(16) describe, until e meets the training goal or the iteration number reaches the upper limit. The parameter of training goal is a predefined threshold. When the training error is less than this threshold, the iteration will stop even if the upper limit of iteration number is not reached:

$$w_{ij}(s+1) = -\eta_1\frac{\partial e}{\partial w_{ij}(s)} + w_{ij}(s), \tag{13}$$

$$w_{jk}(s+1) = -\eta_1\frac{\partial e}{\partial w_{ij}(s)} + w_{jk}(s), \tag{14}$$

$$a_j(s+1) = -\eta_2\frac{\partial e}{\partial a_j(s)} + a_j(s), \tag{15}$$

$$b_j(s+1) = -\eta_2\frac{\partial e}{\partial b_j(s)} + b_j(s), \tag{16}$$

where s is the iteration number and η_1 and η_2 are the defined learning rates for adjusting weights and factors, respectively.

3.2. Genetic Algorithm-Optimized Wavelet Neural Network. The gradient learning method presented above enables the network to gradually adjust along the direction of local improvement, which means inappropriate initial weights and factors can increase the possibility of falling into local optimum or even failing because of the complexity and nondifferentiability of the search space [25]. In this regard, the genetic algorithm (GA) with global search capability is employed to improve the model by optimizing the initial weights and factors in this paper. The specific steps to develop GA-WNN are organized as follows.

Step 1: sample data normalization: normalize the original sample data to an interval of 0–1 using (17) to ensure the quantization uniformity.

$$x_n' = \frac{x_n - x_{\min}}{x_{\max} - x_{\min}}, \tag{17}$$

where x_n' is the normalized value of simple n, x_n is its original value, and $x_{\min}$ and $x_{\max}$ are the minimum and

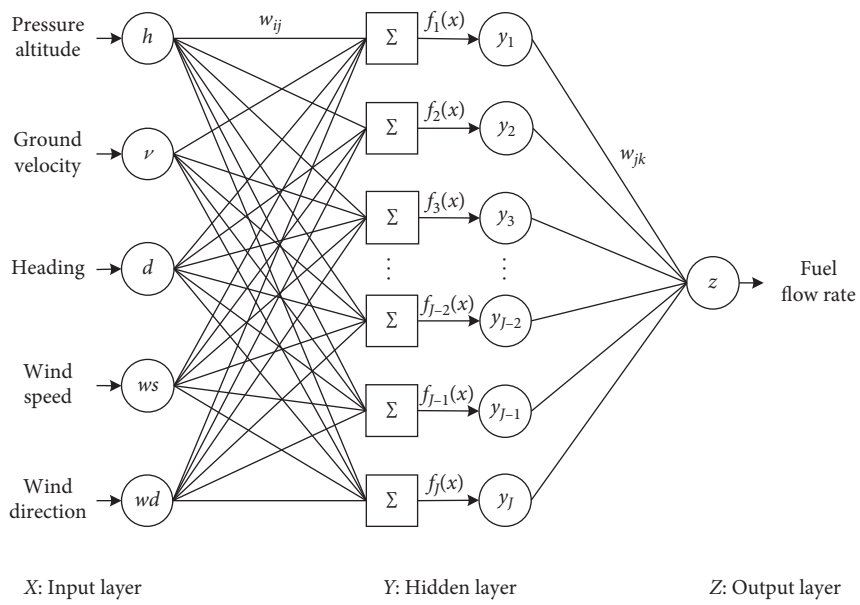


Figure 1: Topology of the WNN for fuel flow rate.

maximum value among all simples of the same category as x_i, respectively.

Step 2: specific architecture determination: use the normalized samples to further determine the architecture of the WNN, including the decision of nodes number in the hidden layer and the selection of wavelet basis functions, on which the network accuracy depends much on prediction accuracy.

Step 3: population initialization: randomly generate an initial set of solutions called population, each of which includes the connection weights for different layers and factors for wavelet basis function.

Step 4: fitness calculation: the fitness F is defined as the mean squared output deviation of all samples in the testing dataset, and the function can be expressed as follows:

$$F = \frac{1}{N}\sum_{n=1}^{N}\left(z_n^{\text{pre}} - z_n^{\text{exp}}\right)^2, \tag{18}$$

where N is the sample number in the testing dataset and z_n^{pre} and z_n^{exp} are the output and actual FFR value of sample n, respectively.

Step 5: iterative evolution: update the population through evolutionary manipulations including selection, crossover, mutation, and retention, and the best individual in each generation is recorded. This step is repeated until a prespecified termination criterion.

Step 6: optimal initial weights and factors acquisition: set the optimal weights and factors according to the individual with the best fitness in Step 5 and assign them to WNN.

Step 7: prediction using GA-WNN: train the model and apply it to new data for reliable prediction results.

3.3. Development of the GA-WNN Model

3.3.1. Sample Processing and Division. Selected flight records data of the medium-weight transport aircraft type, Boeing 737–800, were used in this study to develop and verify the proposed GA-WNN model. The raw data comes from several flights operating in October, 2018, and includes the information of aircraft altitude, Mach number, ground speed, and fuel flow rate of the left and right engine, which is recorded simultaneously by 4 sensors. By averaging the 4 sets, a new set was acquired and the required aircraft cruise data as samples can be extracted from it according to the operation data such as engine speeds and climb rates. Additionally, the FFR(t) in total was calculated by summing that of the left engine and the right one. After normalization, the samples (2,500 in total) were randomly split into training (60%), testing (20%), and validation (20%) sets, allowing a supervised learning process to be performed.

3.3.2. Specific Architecture Determination. As is universally acknowledged, the number of neurons in the hidden layer is a significant parameter to determine the performance of neural networks, and the application of wavelet basis function distinguishes WNNs from traditional BP neural networks and also contributes to a more prospective prediction model. But it is not always better for a larger number in nodes setting and the effect of wavelets differs from each other. Therefore, how many nodes should be designed for the hidden layer and which wavelet basis function is optimal to reduce the gap between predicted outputs and expected ones are critical problems to be considered.

In order to determine the best WNN architecture, comparative tests were carried out by training the WNN for each value in the nodes number range of 6–25 in the hidden layer using different kinds of wavelet basis functions,

respectively. The transfer functions involved in the tests include the original sigmoid function and the following 4 mother wavelets: the *Morlet* wavelet, the *Mexican Hat* wavelet (essentially the *2-order Gaussian* wavelet), the *4-order Gaussian* wavelet, and the *6-order Gaussian* wavelet, which are formulated as follows:

$$\Psi(x) = \cos(1.75x)e^{-(x^2/2)}, \tag{19}$$

$$\Psi(x) = 0.8763\left(1 - x^2\right)e^{-(x^2/2)}, \tag{20}$$

$$\Psi(x) = 1.5\left(3 - 6x^2 + x^4\right)e^{-(x^2/2)}, \tag{21}$$

$$\Psi(x) = \frac{-1}{\sqrt{2\pi}}\left(15 - 45x^2 + 15x^4 - x^6\right)e^{-(x^2/2)}. \tag{22}$$

With the training goal of 0.001, each permutation was executed for 50 runs of 500 epochs with the learning rate of 0.01 for linked weights (w_{ij} and w_{jk}) and 0.001 for wavelet basis function factors (a_j and b_j). Figure 2 shows the average results of the trails.

As is shown in Figure 2, the constructed WNN performs best when 22 neurons are designed in the hidden layer using the wavelet basis function derived from *Morlet* wavelet, and a minimum MSE of about 0.0071 for the testing output can be obtained.

3.3.3. Optimal Initial Weights and Factors. Based on the determined specific architecture of the WNN, the mechanisms of natural selection and genetics were utilized following the steps shown in Section 3.2. For the GA implementation in this paper, the parameters and values were chosen based on the computational results in terms of accuracy during plenty of experiments. Using binary encoding, a population size of 100 was defined and a maximum number of 100 was limited for generation. The evolution manipulations were carried out with the strategies of Roulette-wheel selection, two-point crossover, discrete mutation, and elitism, and the rates for crossover, mutation, and retention were set to be 0.85, 0.01, and 0.85, respectively. The output fitness curve of GA is shown in Figure 3. When the generation number is about 35, there comes a converged solution and a minimum MSE approaching 0.0027, at which time the optimal initial weights and factors can be obtained. And compared with the original minimum MSE of 0.0071 (see Figure 2), a significant improvement of optimization is well-founded.

3.3.4. Performance Measures and Comparation Analysis. Figure 4(a) shows the predicted and actual FFR values for 100 sample exemplars in testing dataset (500 data points in total), it is obvious that there is a good agreement between them for each sample. The output absolute percentage error (APE) of all data points in the testing dataset is counted and illustrated in Figure 4(b); it can be seen that the proportion of output with error lower than 10% is about 70%, and that figure rise to 90% when the error threshold is set to be 20%, preliminarily indicating an acceptable accuracy of the proposed GA-WNN model.

To further verify the reliability of the trained GA-WNN model, the validation dataset was randomly divided into 5 groups (each contains 100 data points) and they were named as G1, G2, G3, G4, and G5, respectively. As can be seen in Figure 5, the predicted FFR values closely match the actual ones in each group, demonstrating a prospective generalizability of the model.

As mentioned previously, traditional BP neural network usually uses the sigmoid function to get nodes output from input. And in its recent application on similar problems such as flight unpredictable fuel calculation [26] and residual fuel prediction [27], a linear function was also employed for this purpose. To assess the advantage of the proposed GA-WNN model for FFR prediction, its performance results were compared with those of the previous models, including BP neural networks using sigmoid (sig) or linear (lin) function to transfer between different layers. Additionally, GA-optimized BP neural networks working with different transfer functions were also compared with using the 5 groups of validation data. Note that all tests were conducted with the gradient method to learn from error so as to correct parameters during iteration. In terms of performance indicators, the mean square error (MSE), mean absolute error (MAE), mean absolute percentage error (MAPE), maximum absolute percentage error (Max-APE), and R^2 were counted and compared. Figure 6 illustrates the detailed performance results using different marked models for each data group. For example, GA-BP (sig-lin) represents a GA-optimized BP model with a sigmoid transfer function between the input and hidden layer and a linear transfer function between the hidden and output layer.

From Figure 6, it can be concluded that GA-WNN shows great advantages in any indicator for each group. Generally, the traditional BP neural networks using different transfer functions are ineffective and still questionable in reliability even after being optimized by GA for their R^2 are all less than 0.73. The WNN performs better than them in all kinds of errors and gets a significant improvement for R^2 valuing about 0.90 in average. Furthermore, compared with the WNN, the GA-WNN model proposed in this paper reduces various errors by more than a half, and R^2 of it reaches a reliable level up to 0.97, which reflects its commendable performance in both accuracy and stability.

4. Numerical Examples

This section employs a set of numerical examples applying the GA-WNN model developed in Section 3.3 to verify the feasibility, reliability, and superiority of the proposed methodology.

4.1. Data Preparation. According to the statistical data of Civil Aviation Administration of China (CAAC), valuing about 30,452, the number of flights operating on the route from Beijing Capital International Airport (ICAO code: ZBAA) to Shanghai Hongqiao International Airport (ICAO

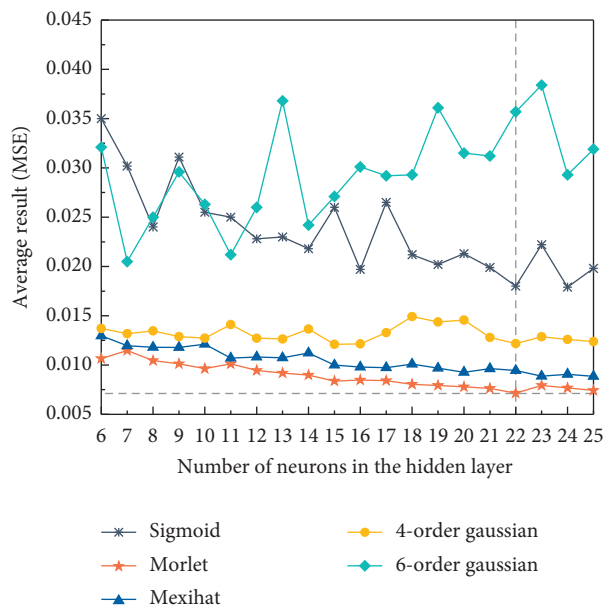


Figure 2: Average MSE of 50 runs using WNN with different architectures.

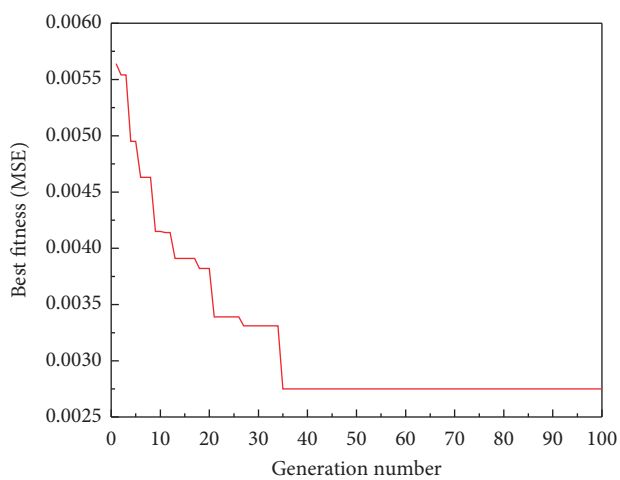


Figure 3: Testing results of GA optimization in terms of best fitness (MSE).

code: ZSSS) during the air season from March 25, 2018, to March 30, 2019, ranks the first among all domestic routes. That means ZBAA-ZSSS is the busiest flight route in China, resulting in the greatest urgency to be evaluated and controlled in regard of aviation environmental impact. Therefore, 9 flights on this route operated by Boeing 737–800 aircraft on October 2, 2019, were taken as examples.

Through the ADS-B tracking data from FlightAware on https://zh.flightaware.com/, the historical all-stage records information (including flight number, time, heading, air pressure altitude, ground speed, longitude, and latitude) of the 9 example flights were collected. Described in terms of Beijing time (UTC+8), their operation durations and profile trajectories are presented in Figure 7, from which the relevant data during cruise phase can be further extracted.

According to the location information of key points on ZBAA-ZSSS shown in Table 1, the meteorological data (including atmospheric pressure, temperature, relative humidity, wind speed, wind direction, etc.) of the sounding spots around the route was gathered from University of Wyoming on http://weather.uwyo.edu/upperair/. As Table 1 shows, there are 10 segments in the route ZBAA-ZSSS in total. However, the first one from ZBAA to ELKUR and the last one from SANSA to ZSSS are for aircraft climbing and descending, respectively; only the middle 8 are for cruise phase and are considered in this paper. Using the spatial interpolation method of inverse distance weighted (IDW), the values of each meteorological parameter at different flight levels corresponding to points on the route were obtained through (23) when the three nearest sounding sites

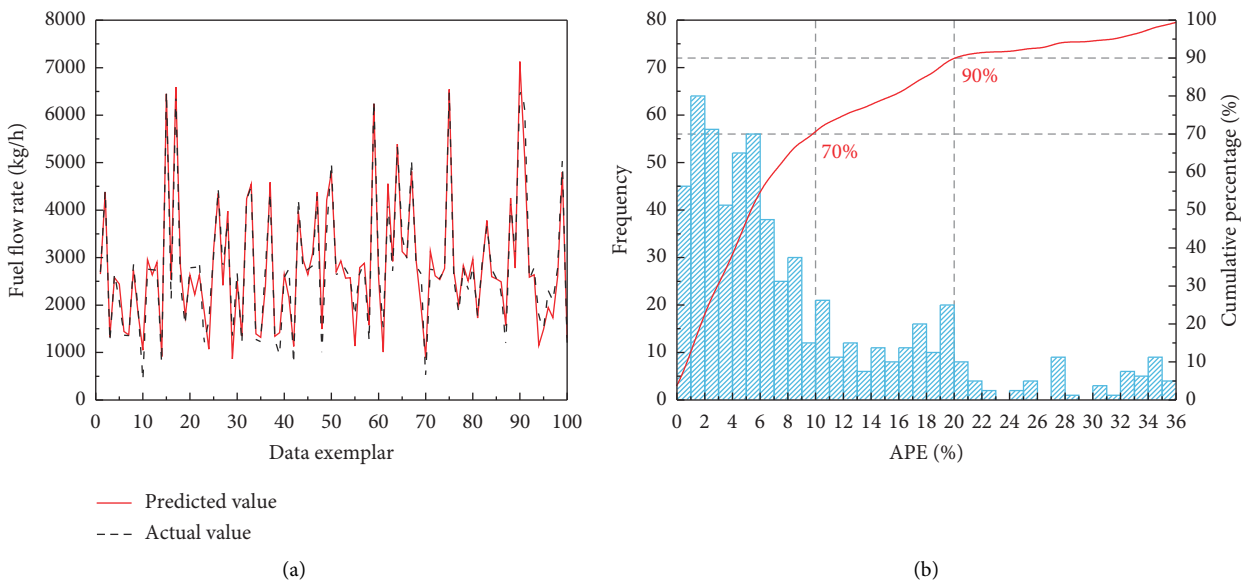


Figure 4: Testing results of GA-WNN model by (a) the comparison of the predicted and actual FFR values for 100 exemplars; (b) the cumulative distribution of absolute percentage error (APE) for 500 data points.

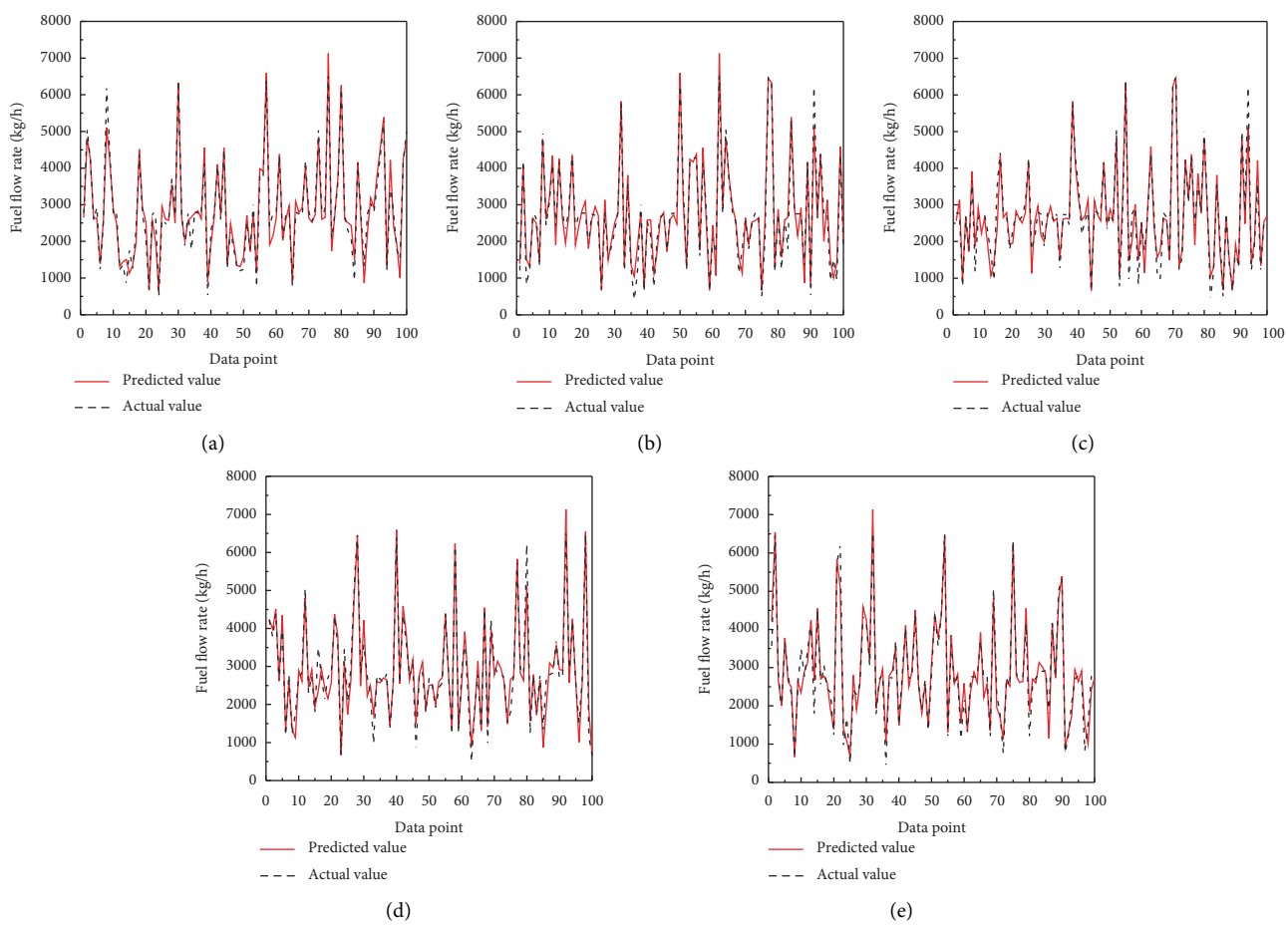


Figure 5: Comparison of the predicted and actual FFR values for the validation dataset of 5 groups including (a) G1, (b) G2, (c) G3, (d) G4, and (e) G5.

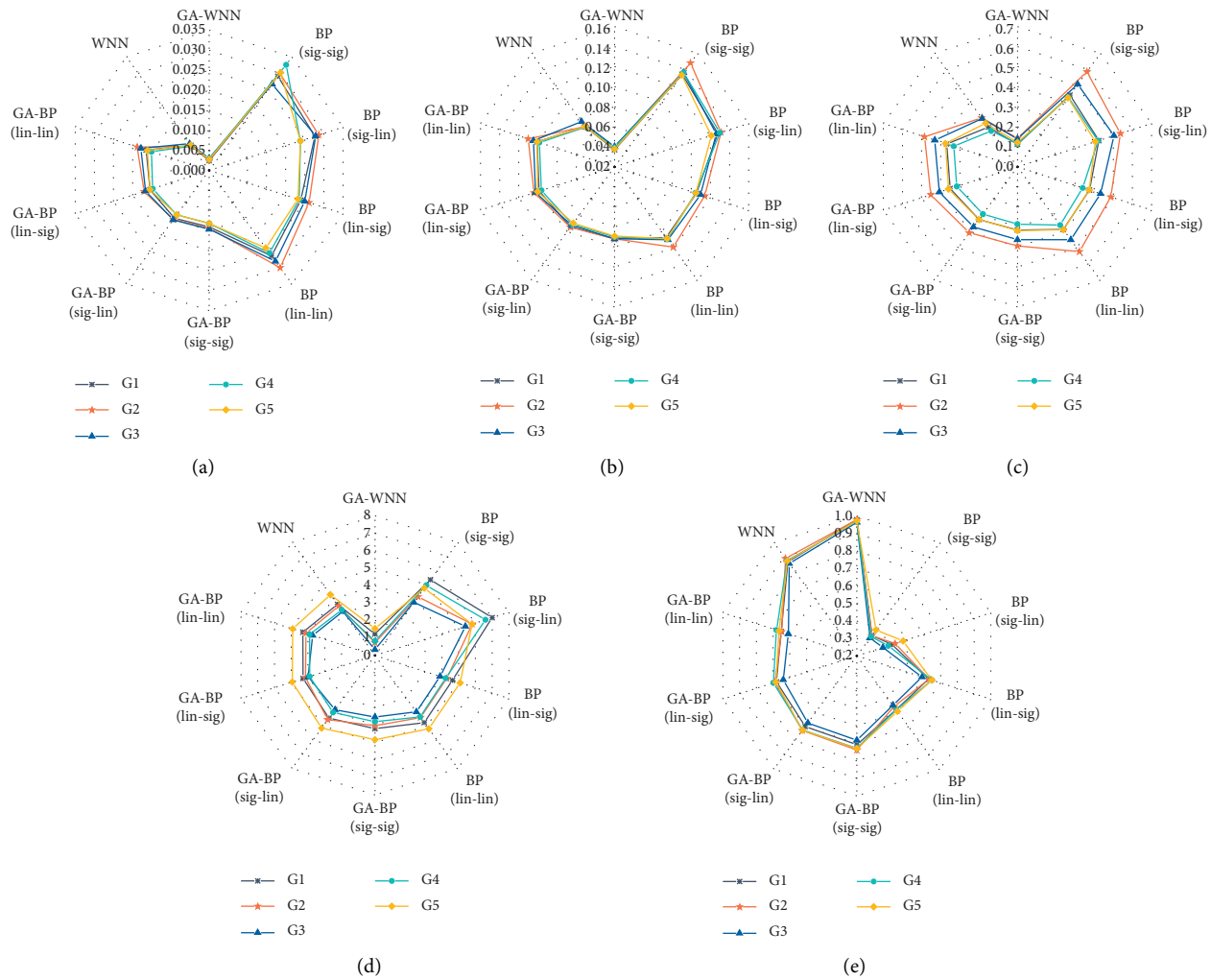


Figure 6: Different models performance using 5 groups of validation data in terms of (a) MSE, (b) MAE, (c) MAPE, (d) Max-APE, and (e) R2.

around were selected as samples and the power exponent was set to be 2:

$$R = \frac{\left(\sum_{m=1}^{3}\left(R_m/l_m^2\right)\right)}{\left(\sum_{m=1}^{3}\left(1/l_m^2\right)\right)}, \tag{23}$$

where R is the needed parameter value on the point to be interpolated; R_m and l_m are the collected value at the sample site m and its distance away from the point to be interpolated, respectively.

After matching the trajectories information with meteorological data by latitude and longitude as well as normalizing through (17), the vector $v_f(t)$ for each flight f at the recorded time t written as (24) can be obtained and thus the input data is ready for the proposed model:

$$v_f(t) = \left\{h_f(t), v_f(t), d_f(t), ws_f(t), wd_f(t)\right\}^T. \tag{24}$$

4.2. Numerical Results. Applying the GA-WNN model, the FFR corresponding to each time point recorded by radar can be predicted and then the fuel consumption can be obtained through integration. Figure 8 shows the fuel consumption of different flights in each of the 8 cruising segments.

Through the total fuel consumption, the CO_2 and NO_x emissions during cruise phase and the resulted surface temperature change for different time horizon of 20, 50, and 100 years can be quantitatively calculated using the functions formulated in Section 2. Table 2 presents the studied flights' results.

4.3. Experimental Analysis. Using the developed methodology in this paper, the detailed results of each flight on all cruise segments can be clearly given once their trajectory data tracked by ADS-B is acquired, while, with fewer need of large amount of operation and state data, the results include the exact temperature change for different time horizons in particular. In terms of evaluation metrics and models, the proposed methodology in this paper is better in investigation depth, result accuracy, and application feasibility compared with previous ones.

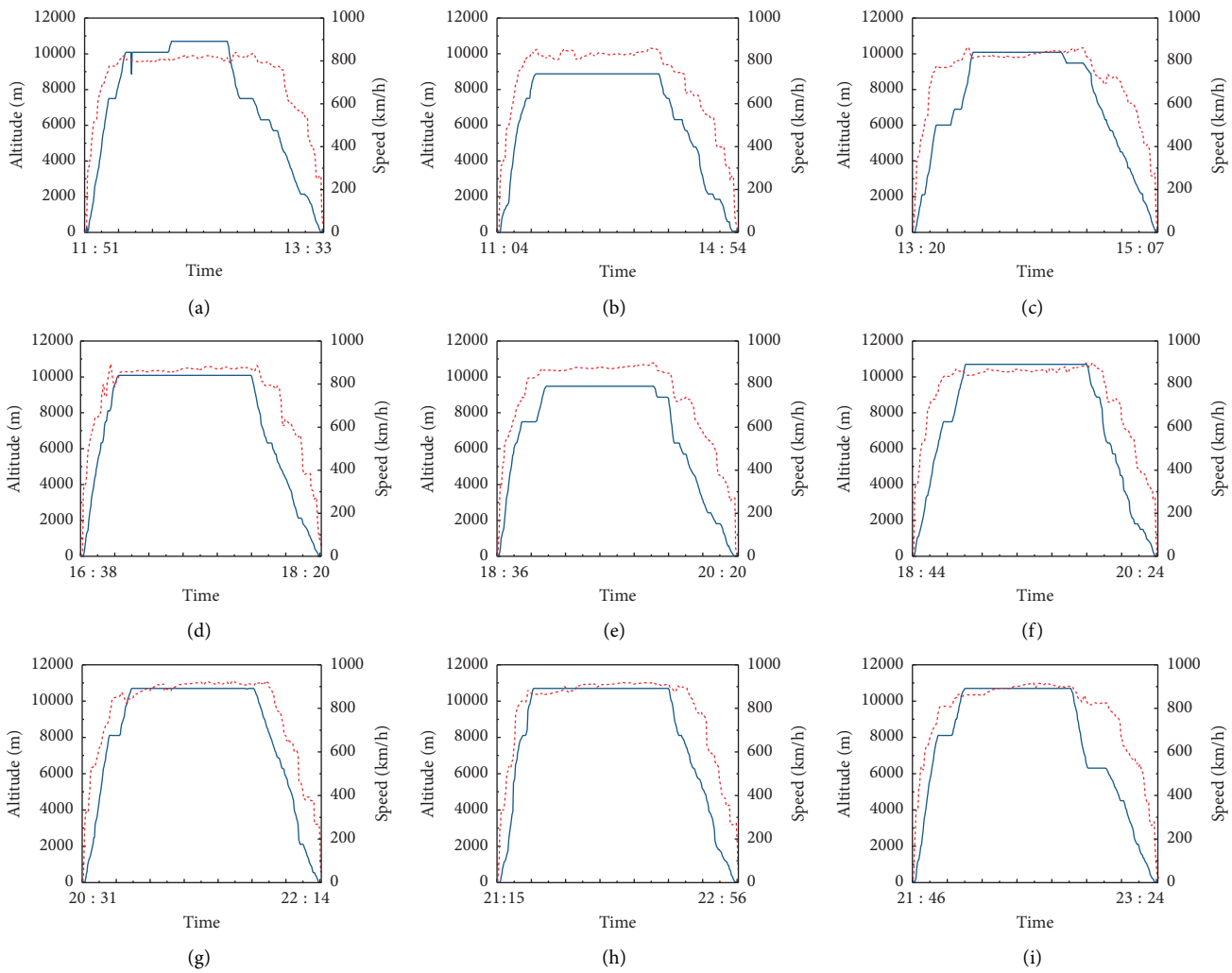


Figure 7: Operation durations and profile trajectories of the studied flights including (a) CCA1557, (b) CES515 R, (c) CXA8178, (d) CCA1549, (e) CSF9108, (f) CCA1885, (g) CSF9102, (h) CHH7603, and (i) CES5158.

Table 1: Location information of key points on the route of ZBAA-ZSSS.

From	To	End latitude (° N)	End longitude (° E)	Segment length (km)
—	ZBAA	40.07	116.60	0.00
ZBAA	ELKUR	38.64	116.67	159.69
ELKUR	OVNUG	38.11	116.70	59.93
OVNUG	GUSIR	37.21	117.04	105.22
GUSIR	ABTUB	36.00	117.37	139.06
ABTUB	UDINO	34.82	117.80	136.96
UDINO	OMUDI	33.97	118.27	105.41
OMUDI	SUBKU	33.20	118.94	106.36
SUBKU	XUTGU	32.46	119.58	100.97
XUTGU	SASAN	31.59	120.32	120.02
SASAN	ZSSS	31.20	121.34	105.94

From Figure 8, it can be seen that cruise flights operating on the same route with various strategies resulted in a gap about 494.69 kg between the minimum, 2706.94 kg, and the maximum, 3201.63 kg, in fuel consumption, leading to different economic benefits and environmental impact. Even for the two flights cruising on similar profile trajectories with analogous speeds and flight levels, such as CSF9102 (see Figure 7(g)) and CHH7603 (see Figure 7(h)), there is also a significant difference in results. Apart from their discrepancy in payload, the atmospheric conditions differing with operating durations contribute a lot to the difference. This effect can be also observed from the different results of the

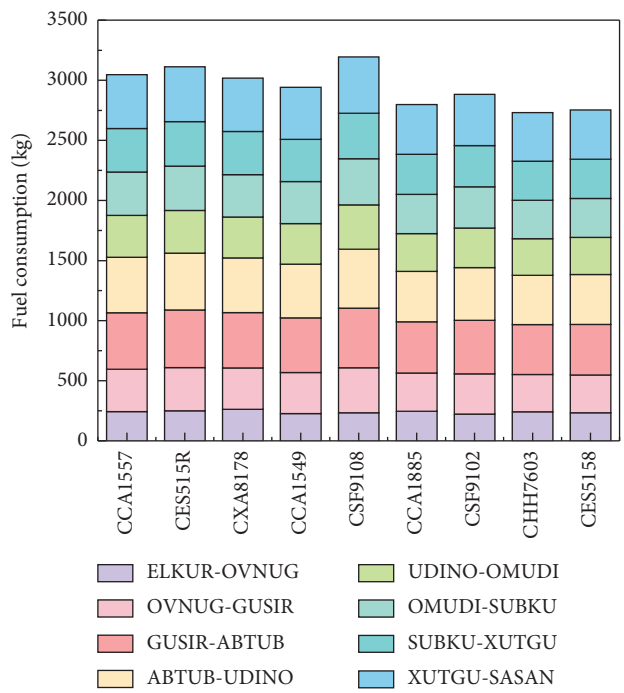


Figure 8: The fuel consumption of studied flights in each segment.

Table 2: Numerical results for studied flights in terms of fuel consumption, gas emissions, and generated greenhouse effect.

Flight number	Fuel consumption (kg)	Gas emissions (kg)		Temperature change [10^{-11} K]		
		E_{CO_2}	E_{NO_x}	ΔT (20)	ΔT (50)	ΔT (100)
(a) CCA1557	3047.40	9614.55	43.56	40.88	7.67	2.49
(b) CES515R	3123.07	9853.29	43.80	41.12	7.72	2.51
(c) CXA8178	3038.95	9587.89	42.40	39.81	7.48	2.43
(d) CCA1549	2942.22	9282.70	40.44	37.98	7.14	2.33
(e) CSF9108	3201.63	10101.14	43.50	40.86	7.69	2.51
(f) CCA1885	2798.59	8829.55	37.29	35.04	6.60	2.16
(g) CSF9102	2873.14	9064.76	37.00	34.78	6.57	2.16
(h) CHH7603	2706.94	8540.40	34.89	32.80	6.19	2.03
(i) CES5158	2752.32	8683.57	36.32	34.13	6.43	2.11

segments that are almost equal in distance and operated by the same flight.

In terms of the generated greenhouse effect shown in Table 2, the global surface temperature change derives from fuel consumption but not always presents a positive correlation with it, indicating a trade-off between economic benefits and environmental contributions must be carefully considered in cruise strategies decisions.

5. Conclusions

In order to analyze the environmental impact and sustainability of modern air transport, a systematic methodology for calculating the greenhouse effect of aircraft cruise is presented and performed in this paper. The main research conclusions are as follows.

(1) The fuel flow rate is modeled with respect to both cruise strategies and wind factors simultaneously and developed using real flight records data based on a supervised learning method, allowing the predicted results more in line with actual operation.

(2) This paper constitutes the first attempt to optimize the initial parameters of WNN by using genetic algorithm in fuel flow-rate modeling. A *Morlet* wavelet and the number of 22 for hidden layer neurons are proved to be preferred and the developed GA-WNN model shows a good agreement between the predicted values and actual ones.

(3) Comparative tests with previous models conducted in this study validate that the developed model is outstanding in both accuracy and stability, with significantly reduced errors for prediction and an acceptable R^2 up to 0.97 in average.

(4) This paper employs 9 flights from ZBAA to ZSSS as numerical examples to implement the proposed methodology for greenhouse effect calculation.

Results of each flight on all cruise segments can be effectively given and clearly compared, from which the factors affecting greenhouse effect generation and how they function can be further figured out.

With less dependency on detailed operation data, more improvements in the model accuracy, and numerical results about the exact temperature change for different time horizons, the proposed methodology in this paper performs better in investigation depth, result accuracy, and application feasibility compared with previous ones. It also supports a feasible, reliable, and superior assessment method for the performance of practical applications, such as aircraft trajectories planning, air traffic management, and operational performance assessment involved with environmental consideration. Future work can be carried out by taking contrails into account as well as exploring an optimization scheme for cruise strategies based on both economic and environmental consideration.

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

The assistance of Chen Wu and Jingfei Liu in data collection is gratefully acknowledged. This work was supported by the National Nature Science Foundation of China (Grant no. 61671237), and the Graduate Open Foundation of Nanjing University of Aeronautics and Astronautics (Grant no. kfjj20200735).

References

[1] Y. Tian, X. He, Y. Xu, L. Wan, and B. Ye, "4D Trajectory optimization of commercial flight for green civil aviation," *IEEE Access*, vol. 8, pp. 62815–62829, 2020.

[2] G. P. Brasseur and M. Gupta, "Impact of aviation on climate," *Bulletin of the American Meteorological Society*, vol. 91, no. 4, pp. 461–464, 2010.

[3] B. Sridhar, N. Y. Chen, and H. K. Ng, "Aircraft trajectory design based on reducing the combined effects of carbon-dioxide, oxides of nitrogen and contrails," in *Proceedings of the AIAA Modelling and Simulation Technologies Conference*, pp. 807–817, Atlanta, GA, USA, August 2006.

[4] Y. Tian, L. L. Wan, B. J. Ye et al., "Cruise flight performance optimization for minimizing green direct operating cost," *Sustainability*, vol. 11, no. 14, Article ID 3899, 2019.

[5] B. Sridhar, N. Y. Chen, and H. K. Ng, "Energy efficient contrail mitigation strategies for reducing the environmental impact of aviation," in *Proceedings of the 10th USA/Europe Air Traffic Management Research and Development Seminar*, Chicago, IL, USA, June 2013.

[6] Q. Xia, H. F. Zuo, and J. L. Yang, "Evaluation of LTO cycle emissions from aircraft in China's civil aviation airports," *Acta Scientiae Circumstantiae*, vol. 28, no. 7, pp. 1469–1474, 2008.

[7] B. Owen, D. S. Lee, and L. Lim, "Flying into the future: aviation emissions scenarios to 2050," *Environmental Science & Technology*, vol. 44, no. 7, pp. 2255–2260, 2010.

[8] Z. Q. Wei, H. Z. Diao, and B. Han, "Research on calculating of aircraft pollution emissions in cruise phase," *Science Technology and Engineering*, vol. 14, no. 19, pp. 122–127, 2014.

[9] A. J. Torija and R. H. Self, "Aircraft classification for efficient modelling of environmental noise impact of aviation," *Journal of Air Transport Management*, vol. 67, pp. 157–168, 2018.

[10] B. Han, W. K. Kong, T. W. Yao et al., "Air pollutant emission inventory from LTO cycles of aircraft in the Beijing-Tianjin-Hebei airport group, China," *Environmental Science*, vol. 41, no. 3, pp. 1143–1150, 2020.

[11] B. Płanda and J. Skorupski, "Methods of air traffic management in the airport area including the environmental factor," *International Journal of Sustainable Transportation*, vol. 11, no. 4, pp. 295–307, 2017.

[12] Z. F. Y. Wang, Y. Tian, L. L. Wan et al., "Progress in the study of environmental impacts of high altitude flight," *Environmental Protection Science*, vol. 53, no. 3, pp. 100–105, 2017.

[13] Eurocontrol, "User manual for the base of aircraft data (BADA)," 2014, https://www.eurocontrol.int/sites/default/files/field_tabs/content/documents/sesar/user-manual-bada-3-12.pdf.

[14] M. Bartel and T. M. Young, "Simplified thrust and fuel consumption models for modern two-shaft turbofan engines," in *Proceedings of the AIAA 7th Aviation Technology, Integration and Operations Conference*, pp. 1450–1456, Belfast, North Ireland, September 2007.

[15] D. A. Senzig, G. G. Fleming, and R. J. Iovinelli, "Modeling of terminal-area airplane fuel consumption," *Journal of Aircraft*, vol. 46, no. 4, pp. 1089–1093, 2009.

[16] E. T. Turgut and M. A. Rosen, "Relationship between fuel consumption and altitude for commercial aircraft during descent: preliminary assessment with a genetic algorithm," *Aerospace Science and Technology*, vol. 17, no. 1, pp. 65–73, 2012.

[17] N. R. J. Lawrance, S. Sukkarieh, and B. Masson, "Using high-frequency data for predicting fuel use of jet transport aircraft," *Journal of Aircraft*, vol. 54, no. 6, pp. 2115–2125, 2017.

[18] J. Liu, *The Aircraft Fuel Estimation Model Based on Flight Data Analysis*, Nanjing University of Aeronautics and Astronautics, Nanjing, China, 2010.

[19] Y. J. Zhang and J. X. Xu, "A novel particle swarm neural network model to optimize aircraft fuel consumption," in *Proceedings of the 4th International Conference on Manufacturing Science and Engineering (ICMSE)*, pp. 3370–3374, Dalian, China, March 2013.

[20] T. Baklacioglu, "Fuel flow-rate modelling of transport aircraft for the climb flight using genetic algorithms," *The Aeronautical Journal*, vol. 119, no. 1212, pp. 173–183, 2015.

[21] Z. Q. Wei and W. X. Zhang, "Research on constructing of matching model between fuel consumption and flight trajectories based on BP neural network," *Flight Dynamics*, vol. 34, no. 6, pp. 25–29, 2016.

[22] X. R. Zhou, *Fuel Flow Precision Estimation Model of Civil Aviation Airplane Based on Deep Learning*, Civil Aviation University of China, Tianjin, China, 2018.

[23] J. S. Fuglestvedt, K. P. Shine, T. Cook et al., "Transport impacts on atmosphere and climate: metrics," *Atmospheric Environment*, vol. 44, no. 37, pp. 4648–4677, 2010.

[24] Y. Lai, S. Easa, D. Sun et al., "Bus arrival time prediction using wavelet neural network trained by improved particle swarm

optimization," *Journal of Advanced Transportation*, vol. 2020, Article ID 7672847, 17 pages, 2020.

[25] A. J. Tallon-Ballesteros, "Metaheuristic algorithm to train product and sigmoid neural network classifiers," *Expert Systems*, vol. 36, Article ID e12383, 2019.

[26] Z. Q. Wei and Y. Hu, "*Unpredictable Fuel Calculation Method Based on BP Neural Network" Flight Dynamics*, vol. 37, no. 6, pp. 7–16, 2019.

[27] R. P. Gu, J. H. Lai, and Z. Q. Wei, "Prediction method of flight residual fuel based on improved BP neural network," *Flight Dynamics*, vol. 1-6, 2020.

Hindawi
Complexity
Volume 2020, Article ID 6309272, 11 pages
https://doi.org/10.1155/2020/6309272

Research Article

Airport Arrival Flow Prediction considering Meteorological Factors Based on Deep-Learning Methods

Zhao Yang, Yifan Wang, Jie Li, Liming Liu, Jiyang Ma, and Yi Zhong

National Key Laboratory of Air Traffic Flow Management, College of Civil Aviation, Nanjing University of Aeronautics and Astronautics, Jiangjun Road No. 29, Nanjing 211106, China

Correspondence should be addressed to Jie Li; lijienuaa@126.com

Received 31 July 2020; Revised 21 September 2020; Accepted 12 October 2020; Published 28 October 2020

Academic Editor: Min Xia

This study presents a combined Long Short-Term Memory and Extreme Gradient Boosting (LSTM-XGBoost) method for flight arrival flow prediction at the airport. Correlation analysis is conducted between the historic arrival flow and input features. The XGBoost method is applied to identify the relative importance of various variables. The historic time-series data of airport arrival flow and selected features are taken as input variables, and the subsequent flight arrival flow is the output variable. The model parameters are sequentially updated based on the recently collected data and the new predicting results. It is found that the prediction accuracy is greatly improved by incorporating the meteorological features. The data analysis results indicate that the developed method can characterize well the dynamics of the airport arrival flow, thereby providing satisfactory prediction results. The prediction performance is compared with benchmark methods including backpropagation neural network, LSTM neural network, support vector machine, gradient boosting regression tree, and XGBoost. The results show that the proposed LSTM-XGBoost model outperforms baseline and state-of-the-art neural network models.

1. Introduction

The airport is the terminal for aircraft taking off and landing. It is also the transferring point for passenger distribution. The daily air traffic flow has strong periodicity and randomness. There are many factors influencing the airport arrival flow, among which the most widely acknowledged are the complex meteorological factors, for example, the change of short-term arrival flow caused by severe weather such as thunderstorm in summer and blizzard in winter, as well as the unfavorable weather conditions that may affect visibility [1, 2]. Real-time and high-precision arrival flow prediction at the airport is of great significance to identify similar patterns, implement passenger evacuation strategy, alleviate airport congestion, and improve air transportation management systems [3–5]. It can also assist passengers to make better traffic mode selection decisions. Therefore, it is necessary to take the meteorological factors into account when forecasting the short-term arrival flow at the airport.

Recently, a series of studies have been conducted regarding the short-term traffic flow prediction based on time-series data. The commonly used methods can be categorized into two groups, including parametric algorithms such as linear regression, time-series models, and Kalman filtering and nonparametric algorithms such as k-nearest neighbor method, support vector regression, deep-learning methods such as neural networks (e.g., convolutional neural network and recurrent neural network), and a combination of these methods [6–11]. The parametric algorithms are easy to implement and can reflect the relation between the independent variables and dependent variable directly, while the nonparametric algorithms, especially the deep-learning method, show superiority with higher prediction accuracy and less computation time for large datasets. For example, Lu et al. proposed a combined method for short-term highway traffic flow prediction based on a recurrent neural network [12]. Asadi and Regan presented a spatiotemporal decomposition-based deep neural network for time-series forecasting with the case of highway traffic flow data from

the Bay Area of California. A multikernel convolutional layer is designed to maintain the network structure and extract short-term and spatial patterns [13]. Li et al. proposed an adaptive real-time prediction model under uncontainable conditions. The model consists of two stages, including an online sequence extreme learning machine with a forgetting factor for noise processing and a hidden Markov model for traffic flow prediction [14].

As compared with highway traffic flow prediction, the short-term prediction of airport arrival flow tends to be more complicated, due to the stochasticity and dynamic nature of air traffic flow considering the various influencing factors such as weather conditions [15–17]. Until recently, the short-term prediction of air traffic flow remains a hot issue. Although different statistical approaches have been used in past studies, each has suggested that there are meaningful relationships between various input variables and traffic flow rate [18–20]. Further development is still needed to advance the predictive aspects of the linkage between airport arrival flow and the input variables including meteorological variables and then to predict future arrival flow using data mining techniques.

The primary objective of this paper is to, first, discover if there are significant relationships between airport arrival flow and various meteorological variables; second, identify which factors can then be used as inputs to estimate airport arrival flow; and third, select an appropriate model that can be used to predict the airport arrival flow with decent performance. To this end, the correlation between historic arrival flow and various features is calculated. Then, a combined Long Short-Term Memory and Extreme Gradient Boosting (LSTM-XGBoost) method is proposed for airport arrival flow prediction. The selected features including meteorological variables are input into the network.

The rest of the paper is organized as follows. Section 2 illustrates the data collection and preparation procedure. Section 3 presents the proposed framework incorporating the long short-term memory neural network and the extreme gradient boosting algorithm components. Section 4 describes the data analysis results by comparing the performance of the proposed method with that of commonly used benchmark methods. Section 5 discusses the conclusions and future works.

2. Data Preparation

To meet the research objective, the airport performance data and various factors required in the data mining procedure are collected. The data sources for analysis can be divided into two categories: flight arrival data and airport meteorological information.

2.1. Flight Arrival Data. This paper selects the flight arrival data of Nanjing Lukou International Airport (NKG) from January 1, 2018, to December 31, 2018, with a total of 113,243 records of information for data extraction and analysis. The specific flight information includes flight ID, aircraft type, departure airport, destination airport, estimated departure time, estimated arrival time, actual departure time, actual arrival time, and status of flight for that day.

The daily flight information is divided into 48 records, with 30 minutes as the time horizon of a record. According to the flight information provided, the flight date, planned and actual arrival time of the aircraft, and the final status of the flight are used to calculate the planned and actual flow data of each time slice of the day. The canceled flights and changed flights on that day are excluded. Figure 1 illustrates the daily arrival and canceled flights in 2018. It can be found that the trend of flight arrivals is periodically fluctuated, while the trend of canceled flights tends to be stochastic and nonscheduled. In addition to the canceled flights, there are also some cases that may cause the difference between the scheduled flight counts and the actual flight counts, that is, change of flight routes, transferring to alternate airports, and missing values. As for the 30 min data records, the difference between the scheduled flight counts and the actual flight counts ranges from 0.014 to 6.803 with a mean value of 2.027, which accounts for 17.56% to 88.47%% with an average of 34.94%.

2.2. Airport Meteorological Information. The airport meteorological information comes from OGIMET [21], which provides local weather conditions. Data from the Meteorological Report of Aerodrome Conditions (METAR) of Nanjing airport in 2018 are collected, including the four-character code of the airport, UTC time, wind direction, wind speed, wind gusts, temperature, dew point temperature, visibility (runway visual range), air pressure, cloud height, cloud cover, humidity, pressure, and weather phenomena such as precipitation, thunderstorm, fog, snowfall, and haze. Variables about some weather phenomena are set as dummy variables. Taking rainfall as an example, 1 indicates the presence of rainfall and 0 indicates no rainfall. The collected METAR messages are summarized. Table 1 presents partial data of the real-time meteorological indicators of Nanjing Lukou International Airport from 10:00 to 14:00 on June 28, 2018, for illustration.

As the METAR information is issued roughly hourly, the linear interpolation method is used to obtain the 30 min granularity meteorological data to match the flow data of 48-time slices per day. Considering that the meteorological information includes not only continuous meteorological factors such as wind speed, temperature, and visibility but also discrete meteorological factors such as rain, snow, and thunderstorm, the piecewise linear interpolation method is used to interpolate the hourly continuous meteorological data, while the weather phenomena are regarded to be consistent in the current one-hour period. Figure 2 illustrates the daily arrival flights as well as the occupied time duration of rain and thunderstorm of NKG in May 2018.

2.3. Data Preprocessing. The collected data are preprocessed by filtering, normalizing, and reconstructing, which effectively improve the convergence speed and prediction

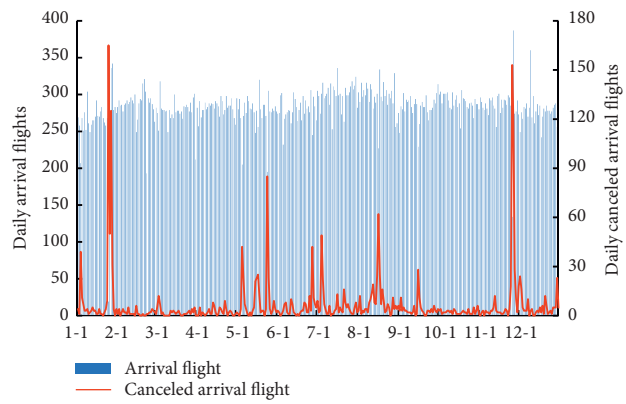


Figure 1: Daily arrival flights and canceled flights in 2018.

Table 1: Partial data of the real-time meteorological indicators.

Time	Meteorological indicators								
	V_{wind}[a]	T[b]	T_{dew}[c]	Visibility	Pressure	Cloud cover	Rain	Thunder	...
10:00	2	25	24	9999	1002	18.37	1	1	...
11:00	4	25	24	7000	1003	21.67	1	1	...
12:00	4	24	24	6000	1002	17.87	1	1	...
13:00	4	24	24	9999	1001	19.27	1	1	...
14:00	7	25	24	9999	1001	19.27	1	0	...
...	...	...	...	...	...	...	...	...	...

[a]Wind speed (m/sec), [b]temperature (°C), and [c]dew point temperature (°C).

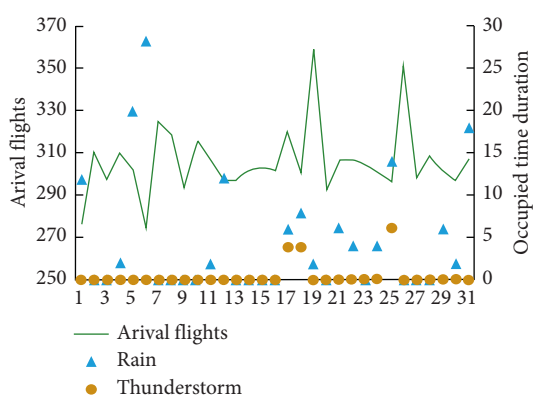


Figure 2: The daily arrival flights and unfavorable weather in May.

accuracy of the model. The final dataset includes one actual inflow as the output variable and twelve features which contain eleven real-time weather features and one planned flow volume as the input variables. All the variables are normalized using the following equation to transform into a dimensionless value ranging from 0 to 1:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}, \quad (1)$$

where x' represents the normalized dimensionless value and x represents the original value. The model is calibrated using data from January to September with a total of 13,104 30 min records and then validated using data from October to December with a total of 4,416 30 min records.

3. Methodology

In this section, a combined LSTM-XGBoost method is constructed for short-term airport arrival flow prediction. The proposed LSTM-XGBoost method contains two components, the long short-term memory neural network and the extreme gradient boosting algorithm. The methods used in each component are briefly discussed.

3.1. The LSTM Method. LSTM is one of the important variants of Recurrent Neural Networks (RNNs). It has been proved that LSTM works well on sequence-based tasks with long-term dependencies. Compared with the traditional artificial neural network, the LSTM network realizes the combination of long-term and short-term memory by setting special structures such as forget gate, input gate, and output gate [22]. In recent years, the LSTM method has been frequently applied in short-term prediction with good performance [23, 24].

As shown in Figure 3, x_t is the input variable and h_t is the output variable at time t. σ and tanh are the activation functions of the network, where σ represents the sigmoid function and tanh is the hyperbolic tangent function. Their role is to introduce nonlinear transformations in neural networks in order to make the network have stronger nonlinear expression capabilities. The data processing procedure of a unit in the LSTM network structure is like this. First, x_t is input together with the output data at the previous time into the network. Then, the long-term memory state variables are selectively remembered through

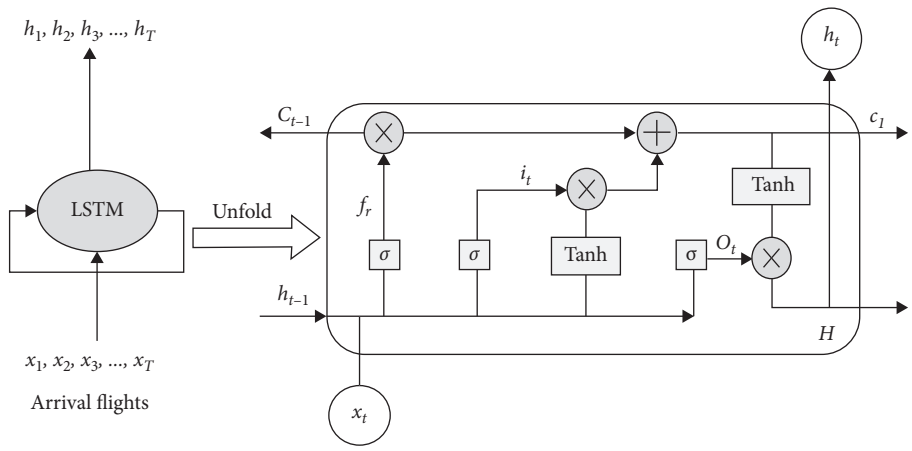


Figure 3: The internal structure of an LSTM cell.

the forget gate, and a new memory state variable is formed by superposing the current state with the long-term state at the previous time through an input gate. Finally, the output variable at time t can be obtained as the long-term memory state variable through the output gate:

$$\mathbf{i}_t = \sigma\left(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{W}_{ci} \odot \mathbf{c}_{t-1} + \mathbf{b}_i\right), \tag{2}$$

$$\mathbf{f}_t = \sigma\left(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{W}_{cf} \odot \mathbf{c}_{t-1} + \mathbf{b}_f\right), \tag{3}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh\left(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c\right), \tag{4}$$

$$\mathbf{o}_t = \sigma\left(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{W}_{co} \odot \mathbf{c}_t + \mathbf{b}_o\right), \tag{5}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh\left(\mathbf{c}_t\right). \tag{6}$$

In equations (2) to (6), W_{*i}, W_{*f}, W_{*c}, W_{*o}, and b_* are learning parameters. $\sigma(*)$ and $\tanh(*)$ are two commonly used nonlinear activation functions.

3.2. The XGBoost Method. The extreme gradient boosting (XGBoost) method is an improved method based on Gradient Boosted Decision Tree (GBDT) proposed by Chen and Carlos (2016) [25]. The salient features of XGBoost which make it different from other gradient boosting algorithms include clever penalization of trees, a proportional shrinking of leaf nodes, newton boosting, and extra randomization parameter. In this paper, the XGBoost method is used to extract features and evaluate relative feature importance. The procedures are presented as follows.

For a given dataset with n samples and M characteristics, represented as $|D| = \{(m_i, y_i)\}\,(m_i \in \mathbb{R}^m, y_i \in \mathbb{R})$, assuming that XGBoost model has K decision trees, the flight flow prediction model is represented as follows:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(m_i), \tag{7}$$

where $\hat{y}_i$ is the predicted value at time i; m_i is the corresponding input variables for $\hat{y}_i$; and f_k is the prediction function corresponding to the kth decision tree, which is defined as follows:

$$f_k(m_i) = \omega_{q(m_i)}, \quad q\colon \mathbb{R}^m \longrightarrow M,\ \omega \in \mathbb{R}^M, \tag{8}$$

where $q(m_i)$ represents the structure function of mapping m_i to the kth decision tree corresponding to the leaf node; ω is the quantization weight vector of the leaf node; and M is the number of leaf nodes in the tree.

The loss function L of the XGBoost algorithm includes error term l and regularization term Ω. The prediction model is learned by minimizing the loss function of the formula. In this paper, the root-mean-square error is selected as error term l, which is defined as follows:

$$L = \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 + \sum_{k=1}^{K} \Omega(f_k). \tag{9}$$

In the formula, the regularization term prevents the model from overfitting.

3.3. The Combined LSTM-XGBoost Method. As mentioned above, the daily air traffic flow has strong periodicity and randomness. Data analysis indicates that there are several peak time of arrival flights, from 8:30 am to 11:00 am, from 12:30 pm to 13:30 pm, and from 17:00 pm to 19:00 pm. The airport arrival flow is influenced by many external factors, among which meteorological factors are commonly recognized that may be significant. The LSTM model has been widely used to deal with time-series problems, which can capture the temporal correlation of time-series data. However, the traditional LSTM lacks the ability to extract the external features that may affect the predicted variables. To this end, this paper proposes an LSTM-XGBoost model, which can well characterize the temporal correlation as well as the influence of external characteristics.

The structure of the LSTM-XGBoost model is shown in Figure 4. The input data of the LSTM cell consists of two parts, including the scheduled flight flow data $z_i^{\cdot}$ and historic flight flow data $x_i^{\cdot}$, constituting the input matrix $X_i^{\cdot}$, where $X_i^{\cdot} \in \mathbb{R}^{2\times T}$; T represents the prediction timestep. After the LSTM layer, the Rectified Linear Unit (Relu) is used as the

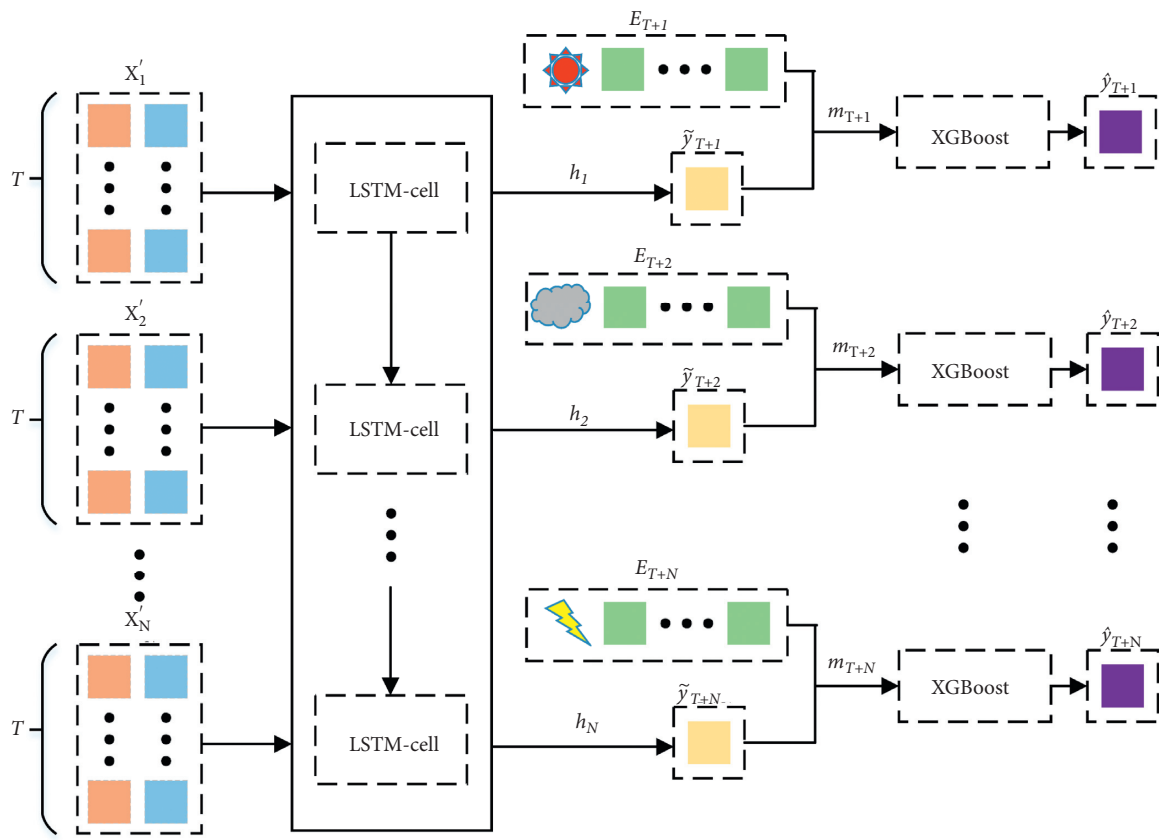


Figure 4: The structure of the proposed LSTM-XGBoost method.

activation function to output the predicted value $\tilde{y}_{T+i}$ at $T+i$ time, which is shown as follows:

$$\tilde{y}_{T+i} = \text{relu}(\omega_h \cdot h_i + b_h). \tag{10}$$

Then, the XGBoost model is used to predict the arrival flow at time $T+i$ from input features m_{T+i}, which incorporates the predicted value from LSTM at time $T+i$ () ($\tilde{y}_{T+i}$) and external meteorological characteristics E_{T+i}:

$$\hat{y}_{T+i} = \sum_{k=1}^{K} f_k(m_{T+i}). \tag{11}$$

3.4. Evaluation Metrics. To evaluate the performance of the proposed model, mean absolute error (MAE), root mean squared error (RMSE), and mean absolute percentage error (MAPE) are calculated for each method, respectively. The equations are shown as follows:

$$\begin{aligned} \text{MAE} &= \frac{1}{n}\sum_{i=1}^{n}\left|\hat{y}_i - y_i\right|, \\ \text{RMSE} &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\hat{y}_i - y_i\right)^2}, \\ \text{MAPE} &= \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|, \end{aligned} \tag{12}$$

where y_i represents the actual value of sample i; $\hat{y}_i$ represents the predicted value of sample i; $\hat{y}_i$ represents the average value of the real data; and n is the sample size.

4. Data Analysis Results

4.1. Correlation Analysis of Input Features. As mentioned above, twelve features are collected and incorporated in the proposed model, including scheduled flights, wind speed, temperature, dew point temperature, visibility, atmospheric pressure at nautical height (QNH), cloud, rain, thunderstorm, fog, snowfall, and haze. To identify the relationship of various factors, the Pearson correlation coefficient (r) between actual arrival flow and the explanatory variables as well as the correlation between different explanatory variables is calculated. The equation is shown as follows:

$$r = \frac{\sum_{i=1}^{n}(y_i - \overline{y})(x_i - \overline{x})}{\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}}. \tag{13}$$

In this formula, x is the independent variable; y is the dependent variable; $\overline{x}$ is the mean of the independent variable; and $\overline{y}$ is the mean of the dependent variable. The Pearson correlation coefficient (r) ranges from −1 to 1, which represents the strength of the linear correlation between two variables. The results are shown in Figure 5.

As shown in Figure 5, it can be found that, besides scheduled flights that are highly related, the actual flights are

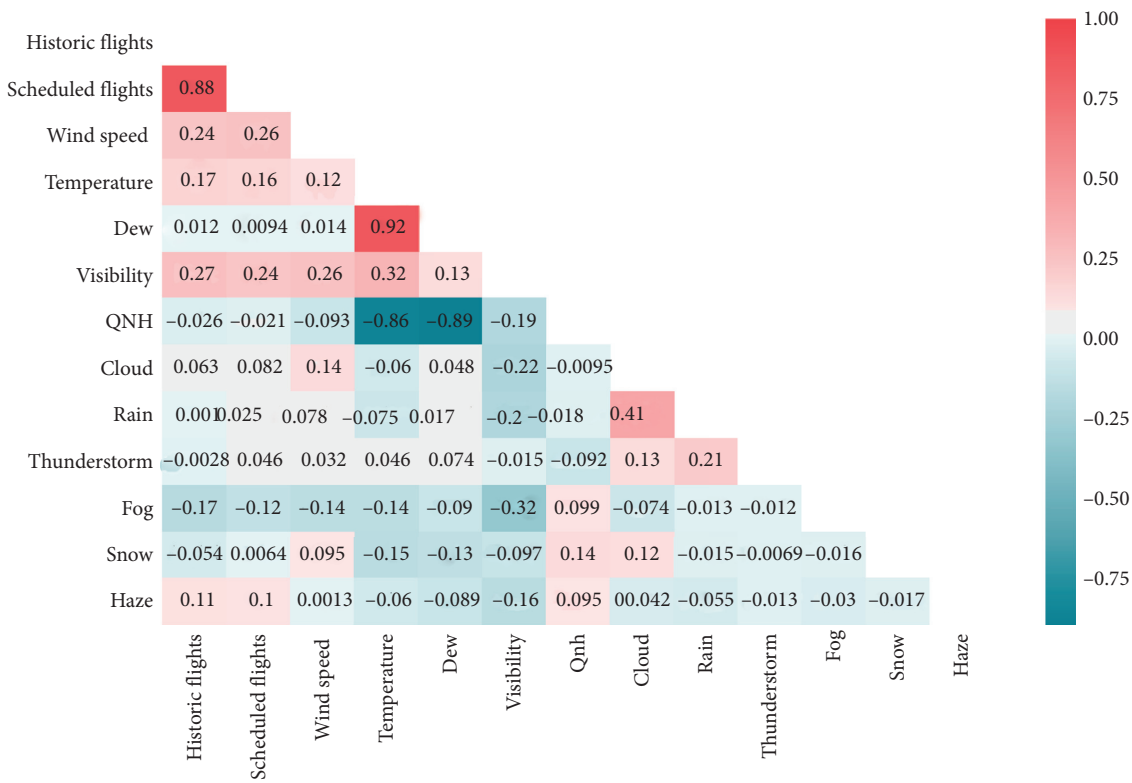


Figure 5: Pearson's correlation coefficient heat map for the input features.

also positively related to visibility, wind speed, and temperature, while negatively related to fog. In addition, the visibility is positively related to temperature, wind speed, scheduled flights, and dew point temperature, while negatively related to fog, cloud, rain, QNH, and haze. It should also be noted that although thunderstorm and snowfall have a weak correlation with the other features with the current data, it does not indicate that these two factors can be excluded from consideration. On the contrary, as rare events, these extreme bad weather conditions may seriously affect the arrival of flights. Considering that, as input features, the temperature is highly positively related to dew point temperature and highly positively negatively to QNH, these two variables (dew point temperature and QNH) are removed from input features in the subsequent models.

4.2. Analysis of Variable Importance. With the selected features, the XGBoost method is applied to identify the relative importance of various variables. The results are shown in Figures 6(a)–6(c) for the 30 min, 60 min, and 120 min prediction time horizon, respectively. Generally, the meteorological variables have a similar impact on the arrival flow for all the three scenarios. The most important influential feature is scheduled flights, which is congenial with common sense. The other two important influential features include temperature and visibility. As for the temperature, it is due to the reason that first, the collected data indicate that, in general, people prefer to travel more in warmer days, except for the traditional holidays. Second, there are more flights in the daytime with higher temperature, as compared with nighttime. Considering the visibility, it is acknowledged that there are visibility requirements for the operation of aircraft. The flights tend to be delayed with poor visibility until it returns to normal conditions.

There are some slight differences for the relative importance of variables of the prediction models with different time periods, which are temperature, followed by visibility, wind speed, cloud, and snow for the 30 min perdition model; visibility, temperature, wind speed, cloud, and snow for the 60 min perdition model; and visibility, temperature, wind speed, snow, and thunderstorm for the 120 min perdition model.

It is also found that the F-scores for the meteorology features are relatively low, while the extreme weather conditions may have strong impacts on the actual flight arrival rate. The collected data indicate that the difference between the actual flow rate and the scheduled flow rate has a higher fluctuation under bad weather conditions. The reason for the small F-scores is that almost all the extreme weather conditions are rare events. The feature importance is generated according to the degree of influence of the feature on the accuracy of the prediction during the process of generating the model. Besides, some of the weather conditions occur at specific time periods during a day. For example, the fog usually appears in the early morning with a lower arrival flow rate. Thus, the calculated importance of the feature will be small according to the collected data. In addition, it is acknowledged that most of the meteorology features are associated with visibility. The impacts of these bad weather conditions are reflected through the perspective of the feature of visibility to a certain extent, rather than the occurrence of snow, thunderstorm, rain, haze, fog, and so on, in terms of dummy variables.

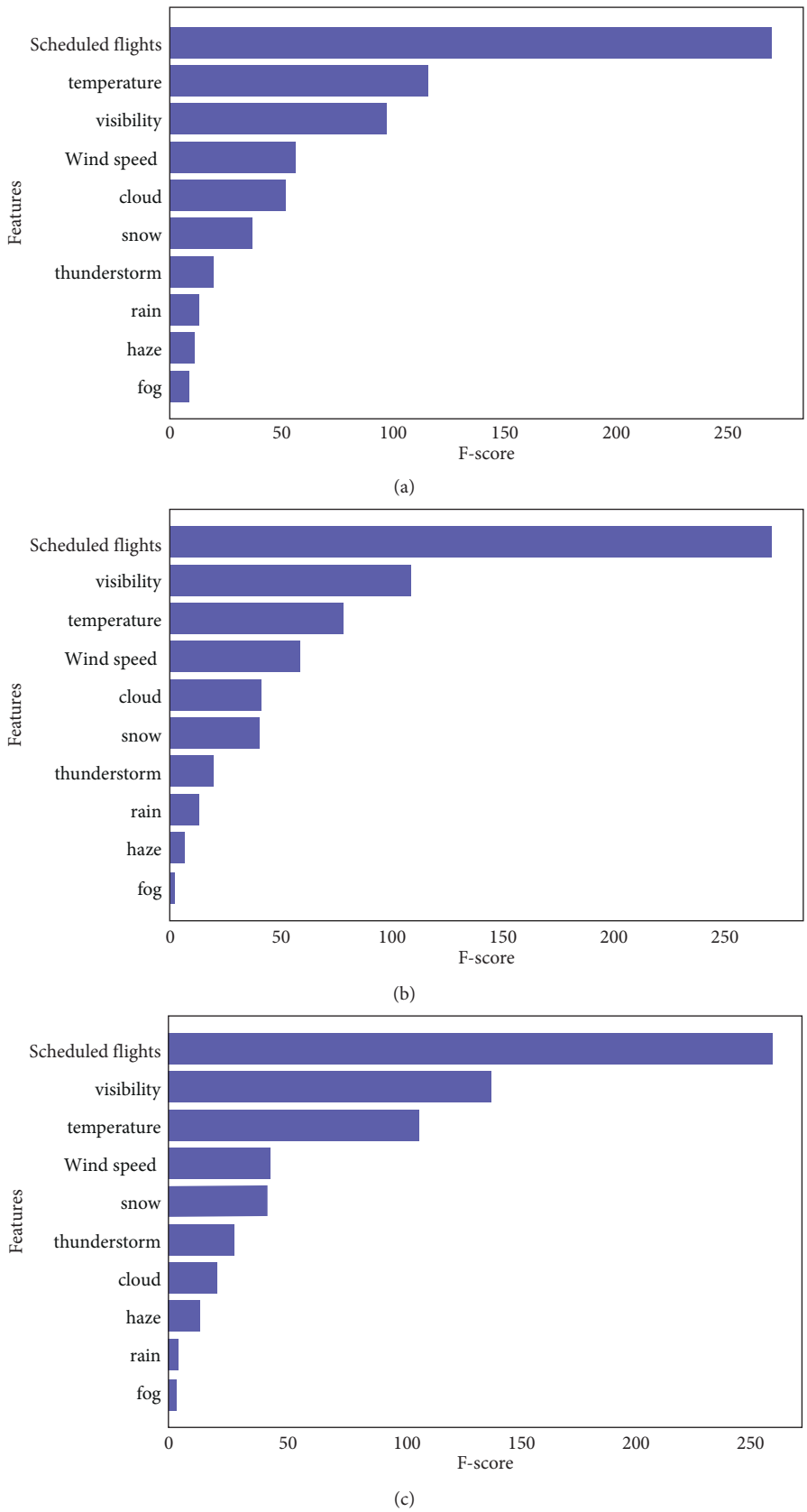


Figure 6: Relative feature importance for 120 min arrival flow prediction. (a) 30 min arrival flow prediction. (b) 60 min arrival flow prediction. (c) 120 min arrival flow prediction.

Table 2: Selection of hyperparameters in the model.

Models	Description of hyperparameters	Input values	The selected optimal hyperparameter
LSTM	Hidden layers	{1, 2, 3, 5, 10}	3
	Number of neurons in each hidden layer	{2, 4, 6, 8, 10, 15, 20, 25, 30}	6
	Timestep	{6, 12, 18, 24, 30, 36, 42, 48}	36
XGBoost	Depth of the tree	{1, 2, 3, 5, 10}	3
	Learning rate	{0.01, 0.02, 0.05, 0.1, 0.15}	0.05
	Number of decision trees	{50, 100, 200, 300, 500}	100

Table 3: Comparison of performances for different methods.

Prediction time horizon (min)	Incorporated features	Model	MAE	RMSE	MAPE[a] (%)
30	Historic and scheduled flights	BP	1.623	2.188	16.657
		SVM	1.825	2.489	23.972
		GBRT	1.628	2.179	16.132
		LSTM	1.580	2.107	14.839
		XGBoost	1.607	2.176	15.050
		LSTM-XGBoost	1.634	2.286	15.005
	Historic and scheduled flights and meteorological variables	BP	1.594	2.148	16.243
		SVM	1.668	2.211	23.474
		GBRT	1.532	2.047	15.257
		LSTM	1.557	2.095	14.515
		XGBoost	1.511	2.023	15.036
		LSTM-XGBoost	1.443	1.989	14.735
60	Historic and scheduled flights	BP	2.408	3.457	13.447
		SVM	2.611	3.714	16.183
		GBRT	2.334	3.301	11.424
		LSTM	2.406	3.254	15.279
		XGBoost	2.307	3.301	11.071
		LSTM-XGBoost	2.319	3.331	11.730
	Historic and scheduled flights and meteorological variables	BP	2.347	3.311	11.557
		SVM	2.447	3.439	14.829
		GBRT	2.324	3.235	11.279
		LSTM	2.365	3.262	17.672
		XGBoost	2.191	3.054	10.783
		LSTM-XGBoost	2.065	2.934	10.834
120	Historic and scheduled flights	BP	3.299	5.171	9.038
		SVM	3.336	5.491	9.372
		GBRT	3.230	4.940	8.693
		LSTM	3.352	5.376	9.359
		XGBoost	3.128	4.933	8.398
		LSTM-XGBoost	3.330	5.387	8.917
	Historic and scheduled flights and meteorological variables	BP	3.166	5.070	8.452
		SVM	3.160	5.187	9.146
		GBRT	3.051	4.782	8.242
		LSTM	3.275	4.751	8.630
		XGBoost	3.039	4.734	8.031
		LSTM-XGBoost	2.889	4.591	7.811

[a]MAPE covers the top 50% highest arrival flow samples in the test dataset.

4.3. Comparison of Prediction Results. With the selected features as inputs, the LSTM-XGBoost model is constructed. The hyperparameters are testified, including hidden layers, number of neurons in each hidden layer, and timestep for the LSTM component as well as the depth of the tree, learning rate, and number of decision trees for the XGBoost component. The input values are shown in Table 2.

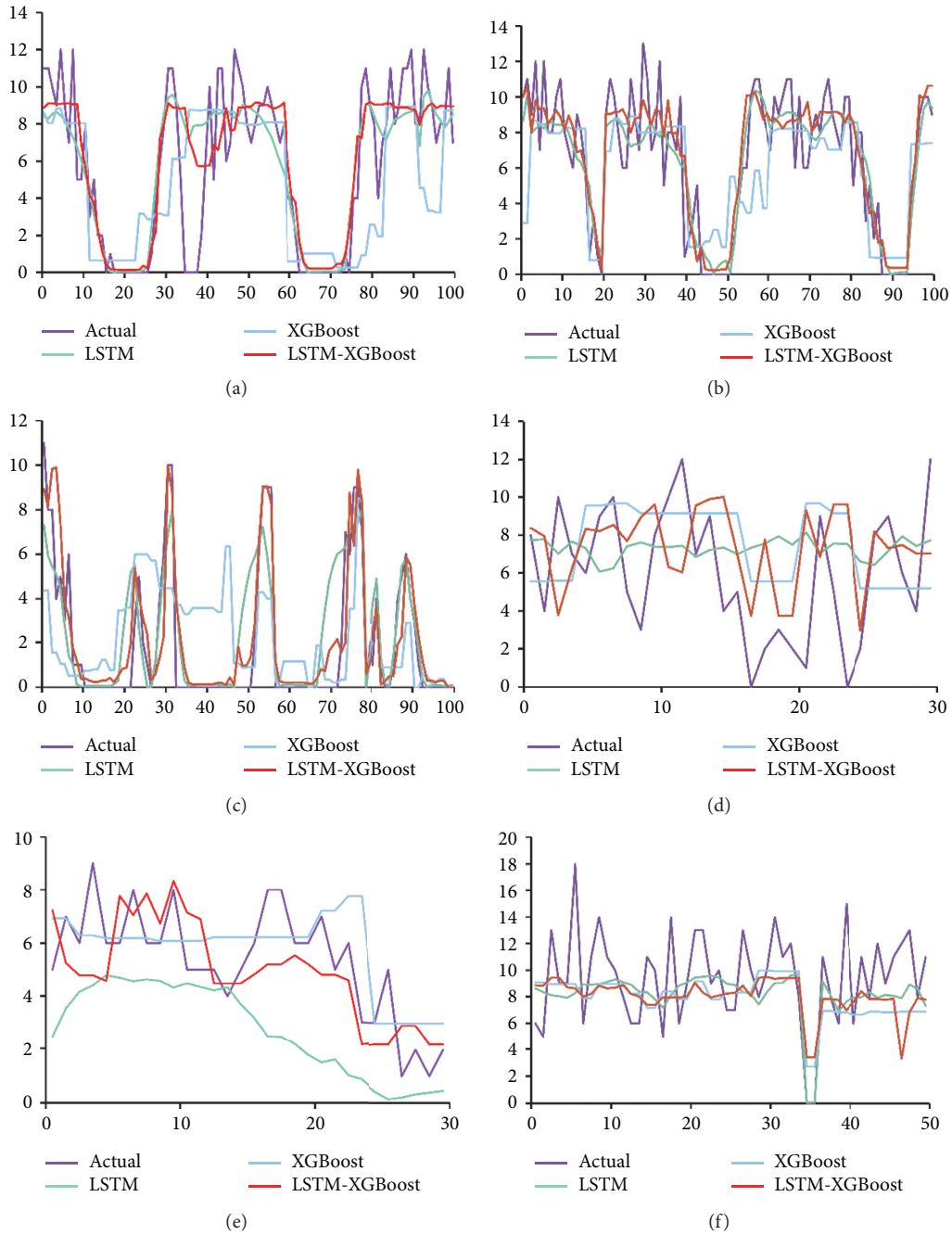


Figure 7: Comparison of model performance under various weather conditions. (a) Sunshine. (b) Rain. (c) Fog. (d) Thunder. (e) Snow. (f) Haze.

To testify the performance of the proposed LSTM-XGBoost model, several benchmark methods are also tested and compared. The selected benchmark methods include backpropagation (BP) neural network, LSTM neural network, support vector machine (SVM), gradient boosting regression tree (GBRT), and XGBoost, which were commonly used in previous studies of short-term traffic flow prediction. The hyperparameters for BP and LSTM are selected in a similar way as that for the LSTM-XGBoost model. All the benchmark methods are trained and tested with the same data and input variables, so as to ensure that the models are comparable. The results are summarized in Table 3.

As shown in Table 3, for each method, six short-term arrival flow prediction models are developed, with 30 min, 60 min, and 120 min as the prediction time level, as well as historic and scheduled flights and historic and scheduled flights together with meteorological variables as input features. Based on the data analysis results, the following findings can be obtained.

First, for each method, MAE, MSE, and RMSE increase sharply with the increase in prediction time horizon, while

MAPE slightly decreases. Specifically, MAE and RMSE are the lowest for the 30 min prediction time horizon, as the two metrics increase with the magnitude of the original arrival flow data, while in terms of MAPE, the model exhibits the best performance for the 120 min prediction time horizon.

Second, for all the five methods, the model performance can be increased by incorporating meteorological variables, especially for the 120 min prediction time horizon, indicating the fact that these factors may have a significant impact on airport arrival flow, especially extreme weather conditions. The improvement is the most prominent for the proposed LSTM-XGBoost method.

Third, the proposed LSTM-XGBoost method generally outperforms all the other machine learning techniques in terms of lower MAE, MSE, RMSE, and MAPE, followed by XGBoost, GBRT, and LSTM. This confirms the superiority and feasibility of the proposed model, which can successfully capture both the temporal features and influencing factors.

To further investigate the performance of the proposed model affected by various meteorological factors, the prediction accuracy of the airport arrival flow for different weather conditions is tested and compared, as shown in Figure 7.

In Figure 7, the x-axis represents the randomly selected samples with 30 min data for each sample. The y-axis represents the number of flights. The prediction results from LSTM, XGBoost, and LSTM-XGBoost methods are compared with the actual data. It is found that the proposed LSTM-XGBoost model outperforms the other two methods for all scenarios. The results further demonstrate the robustness and applicability of the proposed model.

5. Conclusions

This paper proposed a combined Long Short-Term Memory and Extreme Gradient Boosting (LSTM-XGBoost) method for arrival flow prediction at the airport. The traditional Long Short-Term Memory (LSTM) network and the XGBoost model are incorporated by taking both the time-series information and the meteorological features into account. The Pearson correlation coefficients are calculated to describe the strength of the linear correlation between two variables, and the importance of variables is identified. The prediction results are compared with some benchmark methods, including BP, LSTM, SVM, GBRT, and XGBoost. The proposed algorithm improves the accuracy and stability of short-term airport arrival flow prediction.

Even though the proposed LSTM-XGBoost approach has exhibited great potential for short-term prediction of airport arrival flow, several limitations are still needed to be addressed in this study. First, this study is focused on incorporating the meteorological factors in airport arrival flow prediction. As a matter of fact, the real-time airport arrival flow is affected by a series of factors. Future research is still needed to identify the impacts of other significant variables. Second, the paper used the data from Nanjing Lukou International Airport as a case study. Data from other airports can also be applied to further investigate the robustness and applicability of the proposed model, especially those with extreme weather conditions. The authors recommend that future studies could focus on these issues.

Data Availability

The Flight Data.rar file is provided as supplementary materials, containing all the flight arrival data for Nanjing Lukou Airport in 2018. The airport meteorological information is collected from OGIMET (http://ogimet.com/metars.phtml.en).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was sponsored by the Fundamental Research Funds for the Central Universities of China (NS2020046), National Natural Science Foundation of China (51608268, U1933119, and 71971112), and Science and Technology Innovation Project for College Students (2020CX00760 and 2020CX00753).

References

[1] H. Roh, "Development and performance assessment of winter climate hazard models on traffic volume with four model structure types," *American Society of Civil Engineers*, vol. 21, no. 3, 2020.

[2] P. Goswami and S. Sarkar, "An analogue dynamical model for forecasting fog-induced visibility: validation over Delhi," *Meteorological Applications*, vol. 24, pp. 360–375, 2017.

[3] F. G. Habtemichael and M. Cetin, "Short-term traffic flow rate forecasting based on identifying similar traffic patterns," *Transportation Research Part C Emerging Technologies*, vol. 66, pp. 61–78, 2016.

[4] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Short-term traffic forecasting: where we are and where we're going," *Transportation Research Part C: Emerging Technologies*, vol. 43, pp. 3–19, 2014.

[5] K. D. Kuhn, "A methodology for identifying similar days in air traffic flow management initiative planning," *Transportation Research Part C Emerging Technologies*, vol. 69, pp. 1–15, 2016.

[6] D. Han, J. Chen, and J. Sun, "A parallel spatiotemporal deep learning network for highway traffic flow forecasting," *International Journal of Distributed Sensor Networks*, vol. 15, no. 2, 2019.

[7] W. Wei, H. Wu, and H. Ma, "An autoencoder and LSTM-based traffic flow prediction method," *Sensors*, vol. 19, pp. 1–16, 2019.

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

[9] W. Wang, H. Zhang, T. Li et al., "An interpretable model for short term traffic flow prediction," *Mathematics and Computers in Simulation*, vol. 171, pp. 264–278, 2020.

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

[11] Y. Rajabzadeh, A. H. Rezaie, and H. Amindavar, "Short-term traffic flow prediction using time-varying vasicek model," *Transportation Research Part C*, vol. 7, pp. 168–181, 2016.
[12] S. Lu, Q. Zhang, G. Chen et al., "A combined method for short-term traffic flow prediction based on recurrent neural network," *Alexandria Engineering Journal*, 2020, In press.
[13] R. Asadi and A. C. Regan, "A spatio-temporal decomposition based deep neural network for time series forecasting," *Applied Soft Computing Journal*, vol. 87, Article ID 105963, 2020.
[14] W. Li, Y. Ji, and T. Wang, "Adaptive real-time prediction model for short-term traffic flow uncertainty," *American Society of Civil Engineers*, vol. 146, no. 8, 2020.
[15] P. Fleurquin, J. J. Ramasco, and V. M. Eguiluz, "Data-driven modeling of systemic delay propagation under severe meteorological conditions," *Physics*, vol. 84, 2013.
[16] G. Buxi and M. Hansen, "Generating day-of-operation probabilistic capacity scenarios from weather forecasts," *Transportation Research Part C Emerging*, vol. 13, pp. 153–166, 2012.
[17] B. S. Neil and Y. Chen, "Short-term national airspace system delay prediction using weather impacted traffic index," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 2, pp. 657–662, 2009.
[18] Y. Lin, J. Zhang, and H. Liu, "Deep learning based short-term air traffic flow prediction considering temporal–spatial correlation," *Aerospace Science and Technology*, vol. 93, 2019.
[19] H. Liu, X. Zhang, and X. Zhang, "Exploring dynamic evolution and fluctuation characteristics of air traffic flow volume time series: a single waypoint case," *Physica A: Statistical Mechanics and Its Applications*, vol. 503, pp. 560–571, 2018.
[20] H. Liu, X. Zhang, and X. Zhang, "Multiscale complexity analysis on airport air traffic flow volume time series," *Physica A*, vol. 548, 2020.
[21] http://ogimet.com/metars.phtml.en.
[22] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[23] Z. Zhao, W. Chen, X. Wu et al., "LSTM network: a deep learning approach for short-term traffic forecast," *IET Intelligent Transport Systems*, vol. 11, 2017.
[24] Z. Xue and Y. Xue, "Multi long-short term memory models for short term traffic flow prediction," *ICE Transactions on Information and Systems*, vol. E101.D, no. 12, pp. 3272–3275, 2018.
[25] T. Chen and G. Carlos, "XGBoost: a scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, San Francisco, USA, August 2016.

Hindawi
Complexity
Volume 2020, Article ID 5476142, 14 pages
https://doi.org/10.1155/2020/5476142

Research Article

A Marine Object Detection Algorithm Based on SSD and Feature Enhancement

Kai Hu,[1,2] **Feiyu Lu**,[1,2] **Meixia Lu**,[1,2] **Zhiliang Deng**,[1,2] **and Yunping Liu**[1,2]

[1]*College of Automation, Nanjing University of Information Science & Technology, Nanjing 210044, China*
[2]*Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAEET), Nanjing University of Information Science & Technology, Nanjing 210044, China*

Correspondence should be addressed to Kai Hu; nuistpanda@163.com

Received 5 July 2020; Accepted 3 August 2020; Published 30 September 2020

Guest Editor: Zhijie Wang

Autonomous detection and fishing by underwater robots will be the main way to obtain aquatic products in the future; sea urchins are the main research object of aquatic product detection. When the classical Single-Shot MultiBox Detector (SSD) algorithm is applied to the detection of sea urchins, it also has disadvantages of being inaccurate to small targets and insensitive to the direction of the sea urchin. Based on the classic SSD algorithm, this paper proposes a feature-enhanced sea urchin detection algorithm. Firstly, according to the spiny-edge characteristics of a sea urchin, a multidirectional edge detection algorithm is proposed to enhance the feature, which is taken as the 4th channel of image and the original 3 channels of underwater image together as the input for the further deep learning. Then, in order to improve the shortcomings of SSD algorithm's poor ability to detect small targets, resnet 50 is used as the basic framework of the network, and the idea of feature cross-level fusion is adopted to improve the feature expression ability and strengthen semantic information. The open data set provided by the National Natural Science Foundation of China underwater Robot Competition will be used as the test set and training set. Under the same training and test conditions, the AP value of the algorithm in this paper reaches 81.0%, 7.6% higher than the classic SSD algorithm, and the confidence of small target analysis is also improved. Experimental results show that the algorithm in this paper can effectively improve the accuracy of sea urchin detection.

1. Introduction

Autonomous detection and fishing by underwater robots will be the main way to obtain aquatic products in the future, and sea urchins are the main research object of aquatic product detection. To complete the autonomous detection and salvage of underwater robots [1], a series of basic research and scientific problems such as underwater communication [2], underwater positioning [3], information perception [4], target detection and identification [5], and target grasping need to be solved [6], which is an important field of concern for many researchers. Sea urchins are one of the mainstream objects in the current research of aquatic product detection. Detecting and identifying sea urchins in the video is an important prerequisite for salvaging it, and has strong engineering realization value and scientific research significance.

At present, there is no detection algorithm specifically for sea urchins, and target detection methods are based on traditional machine learning and deep learning [7]. The traditional machine learning methods first select some candidate regions on a given underwater image [8], then use methods such as Scale-Invariant Feature Transform (SIFT) and Histogram of Oriented Gradients (HOG) [9] to define features, and finally, use the Support Vector Machine (SVM) [10], Adaptive Boosting(AdaBoost) [11], and other technologies for classification.

However, the traditional target detection method has the following problems: when the underwater light changes rapidly, the algorithm is not effective [12]; when the slow motion and the background color are consistent, the feature pixel cannot be extracted; the time complexity is high; and the noise resistance performance is poor. Compared with

traditional machine learning algorithms, deep learning has the advantages of large data volume, strong scalability, good adaptability, and easy conversion. It can perform end-to-end target detection without specially defined features. It is a powerful method for automatically learning feature representations from data. Based on this, this article uses deep learning methods to carry out related research on sea urchins.

Current target detection methods based on deep learning can be mainly divided into two categories: one is a two-stage target detection algorithm based on candidate regions, such as the Region-based Convolutional Neural Network (R-CNN) [13], Fast Region-based Convolutional Neural Network (Fast R-CNN) [14], and Faster Region-based Convolutional Neural Network (Faster R-CNN) [15]. The other is a one-stage target detection algorithm based on regression, such as SSD (Single-Shot MultiBox Detector) [16] and YOLO (You Only Look Once) [17].

In 2013, Girshick et al. proposed the region-based convolutional neural network R-CNN, and it is a target detection algorithm based on deep learning. The convolutional neural network, which can be applied to image classification tasks, has been successfully applied to image detection tasks. R-CNN target detection achieved an accuracy rate of 53.3% on the Pascal VOC 2012 test set. Compared with the previous best target detection algorithm, the accuracy has been improved by 30%.

In 2015, Ross Girshick improved the previously proposed R-CNN algorithm and proposed Fast R-CNN. Fast R-CNN combines CNN feature extraction and subsequent SVM classification and used a new network to achieve classification and regression. The Fast R-CNN can obtain a one-stage training process through multitask learning, which greatly reduces read and write operations; and after Fast R-CNN sends the whole image into the CNN, the CNN characteristics of different candidate regions in the image are calculated through the mapping relationship so that only one calculation is needed to avoid the problem of repeated calculation. Because of these improvements, Fast R-CNN is 9 times faster than R-CNN in training.

Nonetheless, because Fast R-CNN still uses a selective search strategy, it does not reach the industrial level in speed. In 2016, Ren Shaoqing and Joseph Redmon et al. proposed the Faster R-CNN and YOLO algorithms, respectively. The former was improved on the basis of the Fast R-CNN. The Faster R-CNN built a Region Proposal Network (RPN), this network replaces the selective search method used in the R-CNN and Fast R-CNN, it can train the neural network model end-to-end, and then, accelerate the speed of target detection, so that the detection speed can basically meet the real-time requirements. The latter is a target detection method based on regression. For finding the candidate target box and determining the category of the target, YOLO is carried out on the output layer at the same time, which greatly accelerates the detection speed and reaches 45 fps/s, and it can meet the requirements of real-time target detection.

In the same year, Wei Liu et al. proposed SSD; it combined the anchor mechanism in the Faster R-CNN and the regression idea in YOLO, as the input image feature extraction using a small convolution filter, and the feature of the different scales with different aspect ratio classification prediction. Compared with the Faster R-CNN, the average detection accuracy is basically the same, but the training speed of the model is faster. Compared with YOLO, the detection speed is slightly improved, and the average detection accuracy is increased from 63.4% to 72.1%. Therefore, this article chooses the SSD algorithm to conduct sea urchin detection.

In the preliminary work of applying SSD to sea urchin detection, we conduct experiments on the public data set provided by the National Natural Science Foundation of China Underwater Robot Competition. After data analysis, we found that the existing classic SSD algorithm has a disadvantage of inaccurate detection of small sea urchin targets, and the overall detection performance of the sea urchin has room for further improvement (the AP value of classic SSD is 73.4%).

Considering that, in traditional machine learning, a sea urchin has several important features such as black, round, and spiny, we analyze that deep learning should have no pressure on the extraction of black and round features. Due to the different angles of the thorns and different convolution sizes, it may be that deep learning has the possibility of further improving the detection effect of thorns. Therefore, this paper intends to use feature enhancement methods, trying to enhance the analysis ability of deep learning on the feature learning of spiny, thereby improving the performance of detection and recognition of sea urchins; then, in order to improve the shortcomings of the SSD algorithm's poor ability to detect small targets [18], we use Resnet50 [19] as the basic architecture for network feature extraction to replace the original VGG16 [20] infrastructure of the SSD algorithm and use the feature cross-level fusion idea to improve the feature expression ability and strengthen the semantic information. This idea intends to use 3 fusions to complete the connection between the high-level network and the low-level network, which can expand the scope of the target detection field of view while enhancing the context information of small target prediction.

This article is organized as follows: the second part introduces the background information of the SSD algorithm; the third part describes the improved network algorithm in detail; the fourth part shows the experimental setup, provides the results, and discusses these results; and the fifth part summarizes the full text.

2. Background

The convolutional neural network is a type of feedforward neural network, and it includes convolution calculation and has a deep structure [21]. It has three core ideas: local network connection, convolution kernel parameter sharing, and pooling. The joint effect of local connection and parameter sharing is to reduce the number of parameters, make the operation simple and efficient, and be able to operate on very large data sets. Pooling is to aggregate the characteristics of different locations to obtain lower

dimensions, and it can prevent the problem of overfitting [22]. On this basis, a slight adjustment to the network framework can improve the generalization ability and robustness of the model [23].

The Single-Shot MultiBox Detector (SSD) algorithm is a one-stage algorithm; it is one of the most real-time and advanced target detection algorithms at present. The SSD algorithm has completed the task of classifying and locating the target using only a full-convolution network. The structural framework of SSD is shown in Figure 1. It uses VGG16 as the basic architecture and introduces the design concept of a prior frame. On the basis of VGG16, a new cascaded convolution layer is added to obtain multiscale feature maps to detect the target. All the prediction results are merged together, and the final detection result is obtained by Nonmaximum Suppression (NMS).

The design philosophy of the SSD model is as follows.

2.1. SSD Area Candidate Box. The SSD adopts the method of the multiscale feature map. Region candidate boxes of different sizes and aspect ratios will be set on feature maps of different scales. The regional candidate box definition is calculated as follows:

$$s_k = s_{\min} + \frac{s_{\max} - s_{\min}}{m-1}(k-1), \quad k \in [1, m]. \tag{1}$$

In formula (1), m is the number of feature layers; $s_{\min}$ is the lowest feature map scale (default value is 0.2); $s_{\max}$ is the highest feature map scale (default value is 0.9); and the intermediate feature map scales are evenly distributed. The candidate boxes in the region have different aspect ratios of $a_r \in \{1, 2, 3, 1/2, 1/3\}$. The width and height of the region candidate box are $w_k^a = s_k\sqrt{a_r}$ and $h_k^a = s_k/\sqrt{a_r}$, and for the region candidate box with an aspect ratio of 1, an additional scale $s_k' = \sqrt{s_k s_{k+1}}$ is added. The center coordinates of each region candidate box are $((i+0.5)/w_{fk}, (j+0.5)/h_{fk})$. Of them, w_{fk} is the width of the k feature map, h_{fk} is the height of the k feature map, $j \in [0, h_{fk})$, and $i \in [0, w_{fk})$.

Finally, SSD uses conv4_3, fc7, conv8_2, conv9_2, conv10_2, and conv11_2 as prediction layers. With the deepening of the network, the size of the feature map decreases gradually and the size of the candidate frame increases continuously. Therefore, the shallow feature map is used to detect small targets, and the deep feature map is used to detect large targets.

2.2. SSD Loss Function. During the SSD training process, the target position and category are regressed. The target loss function includes two parts: Location Loss (Loc) and Confidence Loss (Conf). The expression is as follows:

$$L(x, c, l, g) = \frac{1}{N}\left(L_{\text{conf}}(x, c) + aL_{\text{loc}}(x, l, g)\right). \tag{2}$$

In formula (2), N is the number of matches between the regional candidate box and the real box. If $N = 0$, then $L = 0$; x is the matching result of the regional candidate box and the real box of different categories. If it matches $x = 1$, $x = 0$; c is the confidence of the prediction box; l is the position offset information of the prediction box; g is the offset information of the real box and the regional candidate box; and a is the position loss weight usually set to 1.

3. Feature Enhancement

3.1. Multidirectional Edge Feature Enhancement Algorithm. In the preliminary work of applying SSD to sea urchin detection, we found that the detection performance still has room for further improvement (the AP value of classic SSD is 73.4%).

After the analysis, we believe that the sea urchin has several important features such as black, round, and spiny, and we think that deep learning should have no pressure on the extraction of black and round features. Due to the different angles of the thorns and different convolution sizes, it may be that deep learning has the possibility of further improving the detection effect of thorns. Therefore, this paper proposes a sea urchin detection algorithm based on feature enhancement. According to the spiny-edge characteristics of a sea urchin, a multidirectional edge detection algorithm is proposed to enhance the feature, which is taken as the 4th channel of image and the original 3 channels of underwater image together as the input for the further deep learning.

Feature enhancement is to enhance the useful information in the image [24], and it can be a distortion process. The purpose is to improve the analysis effect of the image for the given image application situation [25]. We purposefully emphasize the overall or local characteristics of the image, make the original unclear image clear or emphasize some interesting features [26], expand the difference between the features of different objects in the image, and suppress uninteresting features. The image quality is improved [27], the amount of information is more abundant [28], and the image interpretation and recognition effects are strengthened [29] to meet the needs of some special analyses.

The sea urchin is black overall and has a round shape with thorns. If the edge features of the sea urchin can be effectively extracted, it will have a certain effect on its identification and positioning. The sea urchin spines have a small angle. The detection angle of each operator in the edge detection algorithm is 180 degrees in one direction, the detection angle in 2 directions is 90 degrees, and the detection angle in 4 directions is 45 degrees. It is difficult to accurately identify most sea urchin spines. According to this, this paper proposes 8 directions and 16 directions to improve the detection angle resolution to improve the accuracy of detecting sea urchin edge thorns and, then, use the edge detection channel as the 4th channel to enhance the edge characteristics of the sea urchin.

This paper presents a $5 * 5$ size, 16-direction Prewitt operator, compared with other classic Sobel operator 2-direction, Laplace operator unidirectional, Prewitt operator 2-direction, Prewitt operator 4-direction, and Prewitt operator 8-direction. The results are shown in Figures 2(b)–2(g) and 3(b)–3(g).

Figures 2(a) and 3(a) are two sea urchin maps randomly selected from the public data set provided by the National

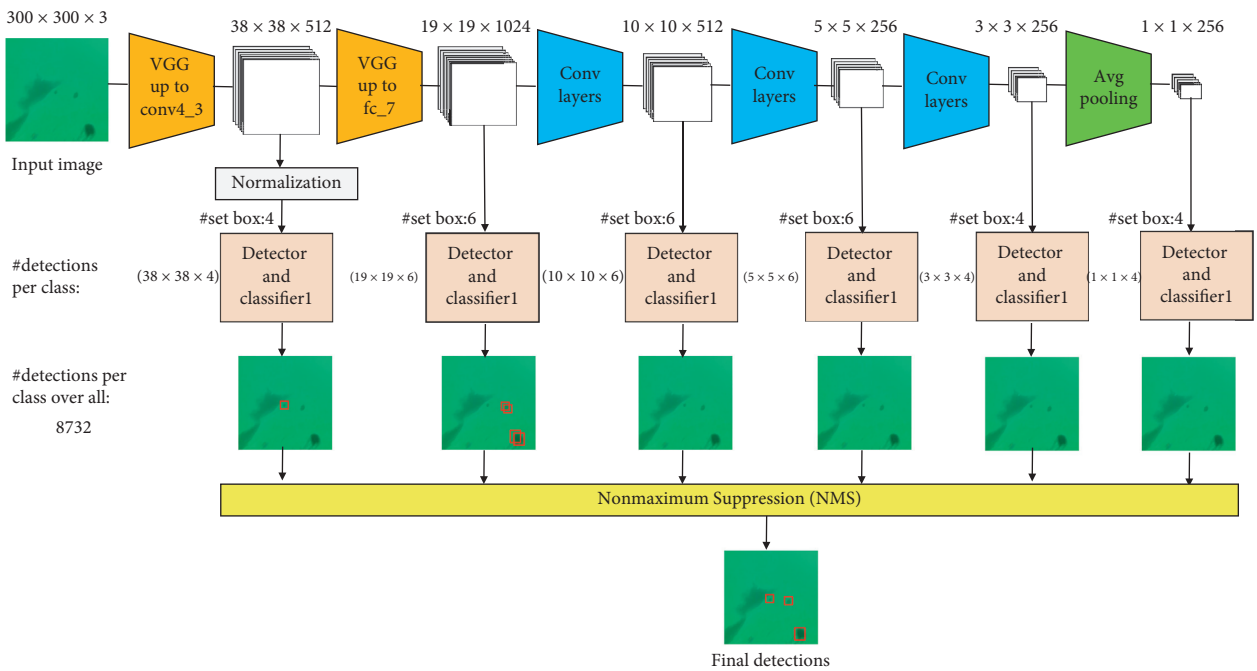


Figure 1: SSD structure frame.

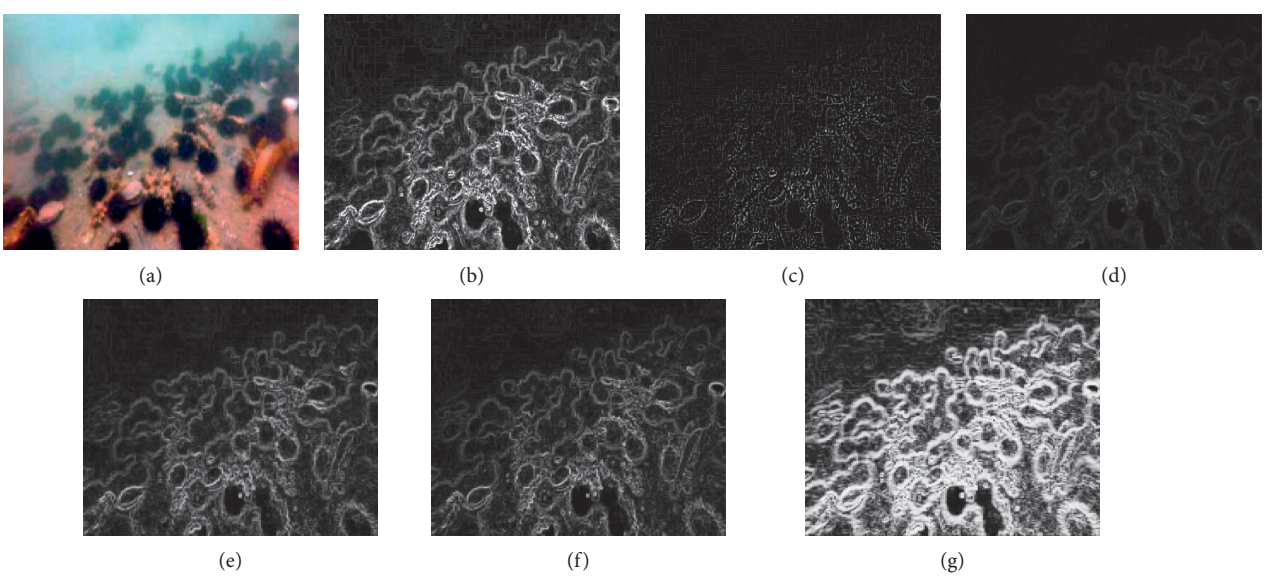


Figure 2: (a) The image of underwater sea urchins; (b) the image of Sobel operator 2-direction edge detection; (c) the image of Laplace operator single-direction edge detection; (d) the image of Prewitt operator 2-direction edge detection; (e) the image of Prewitt operator 4-direction edge detection; (f) the image of Prewitt operator 8-direction edge detection; and (g) the image of Prewitt operator 16-direction edge detection.

Natural Science Foundation of China Underwater Robot Competition. Figure 2(a) is a scene of a large number of small target sea urchins. Figure 3(a) is a scene of multi-category aquatic products interference, and it is more representative. Comparing the image edge extraction result maps in different scenes, it is possible to compare the effect differences of the three operators. Figures 2(b)2(g) and 3(b) 3(g) are the effect diagrams after performing edge detection for Sobel, Prewitt and Laplace operators in Figures 2(a) and 3(a). It can be found that the Laplace operator single-direction edge detection effect is not as good as the Sobel operator 2-direction, Prewitt operator 4-direction, and Prewitt operator 8-direction, but the edge spines of sea urchin can be better extracted. In contrast, the Prewitt

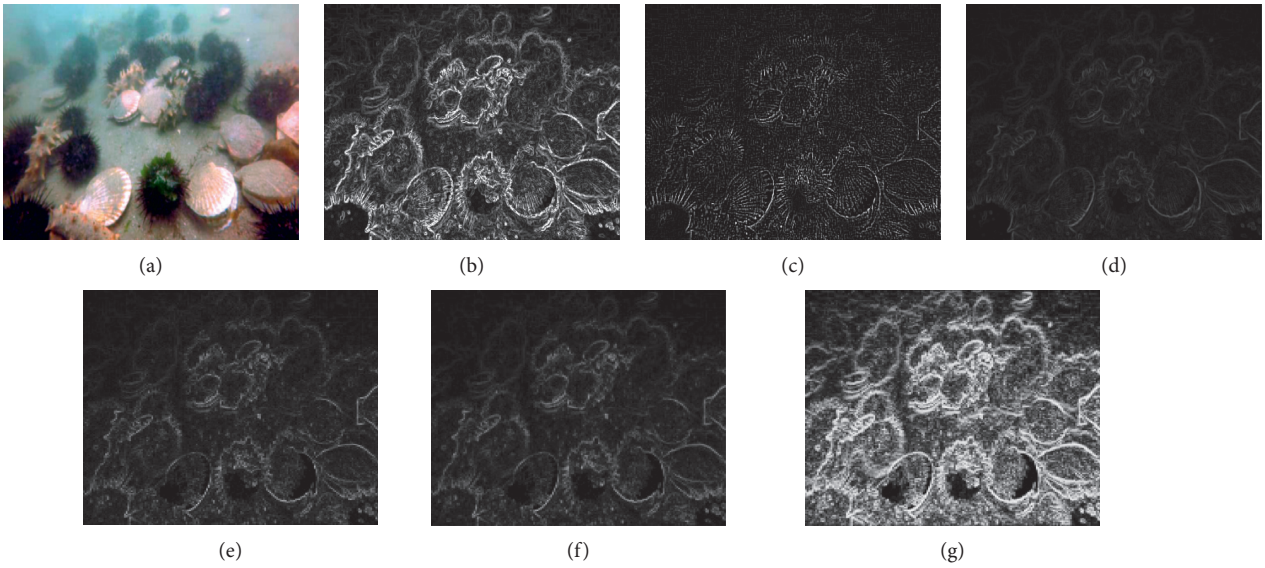


Figure 3: (a) The image of underwater sea urchins; (b) the image of Sobel operator 2-direction edge detection; (c) the image of Laplace operator single-direction edge detection; (d) the image of Prewitt operator 2-direction edge detection; (e) the image of Prewitt operator 4-direction edge detection; (f) the image of Prewitt operator 8-direction edge detection; and (g) the image of Prewitt operator 16-direction edge detection.

operator 2-direction of edge detection effect is poorer, it is hard to find the edge information of the sea urchin, but the noise is less. Among them, the Prewitt operator 16-direction edge detection has the best effect and the edge gray value is high, it can detect the edge information of the sea urchin better, but the noise is larger, while the SSD algorithm has a strong ability to suppress noise, so the noise is not considered for the time-being problem.

The visualization effect of Prewitt operator 16-direction edge detection is the best. The multidirectional edge detection process takes Prewitt operator 16-direction as an example, as shown in feature enhancement program. Taking the output image as the 4th channel, it can be clearly seen that the thorny area on the edge of the sea urchin is highlighted, and the background area is almost white, so that the background and urchin can be better segmented.

Feature enhancement program:

Step 1. Read the image

Step 2. Change the color map into grayscale

Step 3. Input Prewitt operator 16-direction stencil, such as $x1$, $x2$, $x3, \ldots, x16$; the gradient images are obtained by using the Prewitt gradient operator in 16 directions and converted into CV_8UC1

Step 4. OTSU binarization is performed on the converted gradient image to obtain the binarization image

Step 5. Carry out and operation on 16 binary images according to corresponding pixels

Step 6. Conduct multiple iterations of expansion and corrosion operations on the binaries

Step 7. Contour search and fill for small area blocks

Step 8. Background corrosion

Step 9. Output the mask diagram

3.2. Feature Cross-Level Fusion Mode. In the preliminary work of applying SSD to the detection of sea urchin, we found that the detection performance has room for further improvement (the AP value of classic SSD is 73.4%).

Through in-depth analysis, we think the main problem is that some sea urchins are small targets, and the traditional SSD algorithm has a relatively poor detection effect on small-sized objects. The feature map representation capability of shallow extraction is not strong enough. In this way, there will be misdetection and missed detection of small targets of the sea urchin. According to this, in order to improve the characteristics of the poor recognition ability of the SSD algorithm for small targets, the improved SSD algorithm draws on the idea of the residual network and uses Resnet 50 instead of VGG16 as the basic framework of the network. The network architecture is shown in Figure 4. Deepening the neural network by learning the residuals can avoid the problems of overfitting and the disappearance of the network gradient, learn more abstract texture features and semantic features, and strengthen the expression ability of features, so as to improve the ability of target classification and location. At the same time, a feature cross-level fusion method is proposed to improve the feature expression ability and strengthen the semantic information and further improve the problem of poor detection ability of the SSD algorithm for small targets.

In Figure 4, three fusion modules are used to complete the connection between the high-level network and the low-level network. The context information of small target prediction is enhanced, and the field of view of target detection is expanded. The fusion feature of fusion module 1, that is, the skip connection of res2_3 and res5_3, is fed into conv6, as shown in Figure 5. In order to fuse the feature maps of res2_3 and res5_3, the feature maps of res5_3 need to be upsampled. First, the res5_3 feature map is upsampled

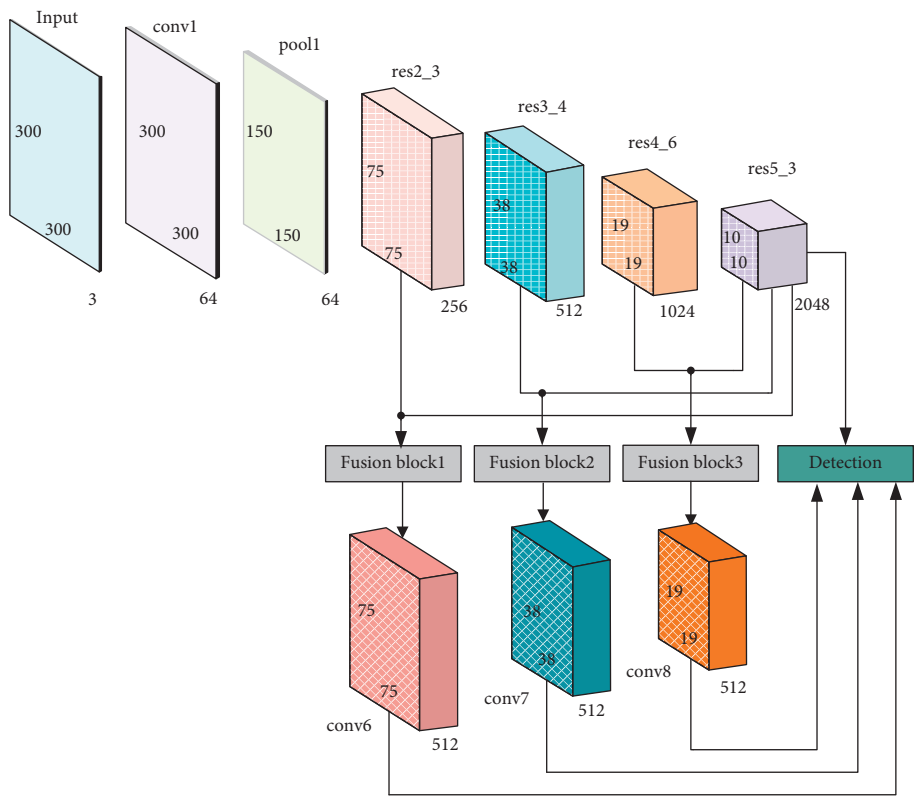


Figure 4: Network architecture diagram.

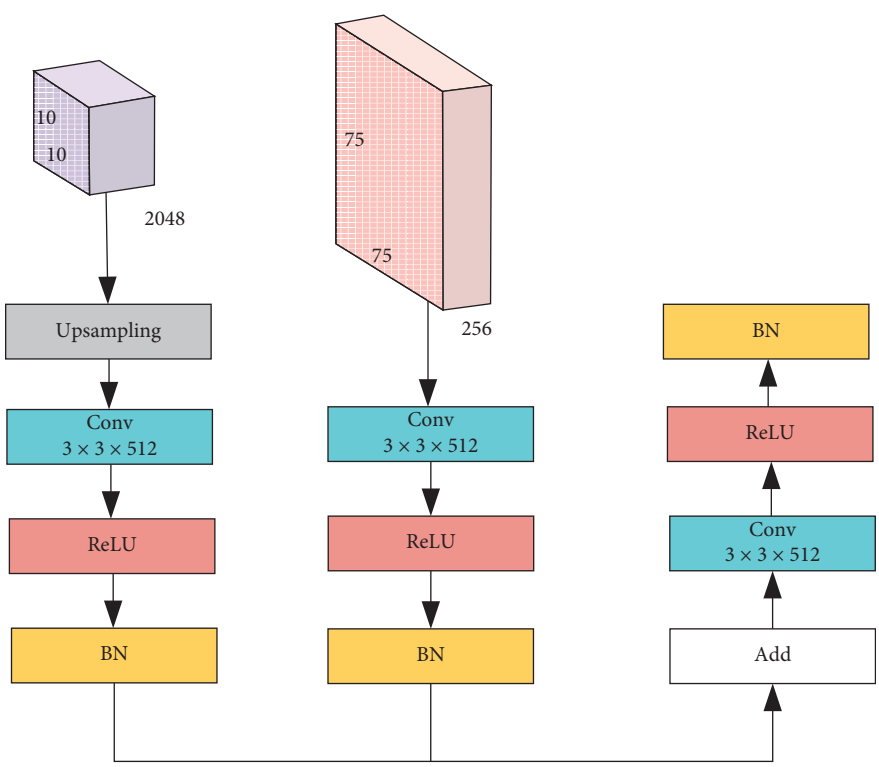


Figure 5: Features of cross-level fusion modules.

to the same size as res2_3 by interpolation and upsampling. The output of the upsampling is mapped to the modified activation function layer (Rectified Linear Unit (ReLU)) [22] through a convolution layer with a convolution kernel of 3×3. Then, go through the L2 regularization layer for normalization (Batch Normalization (BN)). Res2_3 is

directly mapped to the Relu activation function layer through a 3×3 convolution kernel map and, then, input to the L2 regularization layer. The output between the two partitions is summed and passed to the Relu layer after merging. Finally, $256 \cdot 3 \times 3$ convolution kernels are used to ensure that the detected features are distinguishable, and the fusion function is realized after a Relu layer. The fusion module 2 is the feature fusion of res3_4 and res5_3 jump-level connection to conv7. The fusion module 3 is the feature fusion of res4_6 and res5_3 jump-level connection to conv8. Finally, the four feature maps of conv6 (75×75), conv7 (38×38), conv8 (19×19), and res5_3 (10×10) after feature fusion are sent to the prediction module for prediction.

3.3. Overall Improvement Process. Figure 6 is the overall flow chart of underwater image detection, which simply describes the calculation and operation process of the improved SSD algorithm. After analysis, we believe that the sea urchin has several important features such as being black, round, and spiny, among which deep learning should have no pressure on the extraction of black and round features. Due to the different angles of the thorns and different convolution sizes, it may be that deep learning has the possibility of further improving the detection effect of thorns. Therefore, this paper proposes a sea urchin detection algorithm based on feature enhancement. According to the spiny-edge characteristics of sea urchin, a Prewitt operator 16-direction edge detection algorithm is proposed to enhance the feature, which is taken as the 4th channel of image and the original 3 channels of underwater image together as the input for the further deep learning.

At the same time, some sea urchins belong to small targets, and the traditional SSD algorithm has a relatively poor detection effect on small-sized objects. According to this, in order to improve the characteristics of the SSD algorithm's poor ability to recognize small targets, the improved SSD algorithm draws on the idea of residual network and uses Resnet 50 and replaces VGG16 as the basic framework of the network, and it can avoid the problems of overfitting and the disappearance of the network gradient. It adopts the feature cross-level fusion idea to improve the feature expression ability and strengthen the semantic information. Finally, the feature map is sent to the trained model for prediction, and the sea urchin detection result map is obtained.

4. Experimental Analysis

CPU: Inter i7-9700k, memory: 16G DDR4, GPU: Nvidia Geforce GTX2080Ti, operating system: 64-bit Ubuntu 16.04, and the experimental framework is Pytorch open source framework. Stochastic gradient descent is used as the learning rate, the initial learning rate is 0.001, and the learning rate is reduced by 10 times when the number of iterations is 100, 150, and 200 cycles; the momentum is set to 0.9, the weight attenuation coefficient is set to 0.0001, and the training batch size is 32, training 300 cycles.

This article uses the public data set provided by the National Natural Science Foundation of China Underwater Robot Competition [30] as the training set and test set. The public data set contains 3000 training pictures and 800 test pictures, a total of 3800 pictures, and the picture size is uniformly $300 * 300$. Among them, the sea urchins vary in size and type. The 3800 pictures are all underwater images, with different lighting conditions, complex backgrounds [31], and varying degrees of occlusion. The detection of sea urchins and other targets in this article is actually a two-category problem. The ultimate goal is to correctly detect all sea urchins and reduce the missed detection rate and the false detection rate.

In order to better evaluate the model, we set TP (True Positives) to represent the real class, which is the positive sample predicted correctly by the model, and FP (False Positives) to represent the true negative class, which is the positive sample predicted by the model to be negative. FN (False Negatives) represents a false negative class, which is the negative sample predicted by the model as positive, and TN (True Negatives) represents a true negative class, which is the negative sample predicted by the model as negative. The formulas of accuracy and recall rate are as follows:

$$\begin{aligned} \text{precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}}, \\ \text{recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}}. \end{aligned} \tag{3}$$

For each category of target detection, a check line-recall rate (P-R) curve can be obtained, and the accuracy under the curve is Average Precision (AP). For this task, there is only one type, so AP is mAP (mean Average Precision). The AP formula is as follows:

$$\text{AP} = \int_0^1 P(R)\mathrm{d}R. \tag{4}$$

4.1. Analysis on the Rationality of the Multidirectional Edge Feature Enhancement Algorithm. In the preliminary work of applying SSD to sea urchin detection, we found that the detection performance still has room for further improvement (the AP value of classic SSD is 73.4%).

After analysis, we believe that the sea urchin has several important features such as being black, round, and spiny, and we think that deep learning should have no pressure on the extraction of black and round features. Due to the different angles of the thorns and different convolution sizes, it may be that deep learning has the possibility of further improving the detection effect of thorns. Therefore, this paper proposes a sea urchin detection algorithm based on feature enhancement. According to the spiny-edge characteristics of sea urchin, a multidirectional edge detection algorithm is proposed to enhance the feature, which is taken as the 4th channel of image and the original 3 channels of underwater image together as the input for the further deep learning.

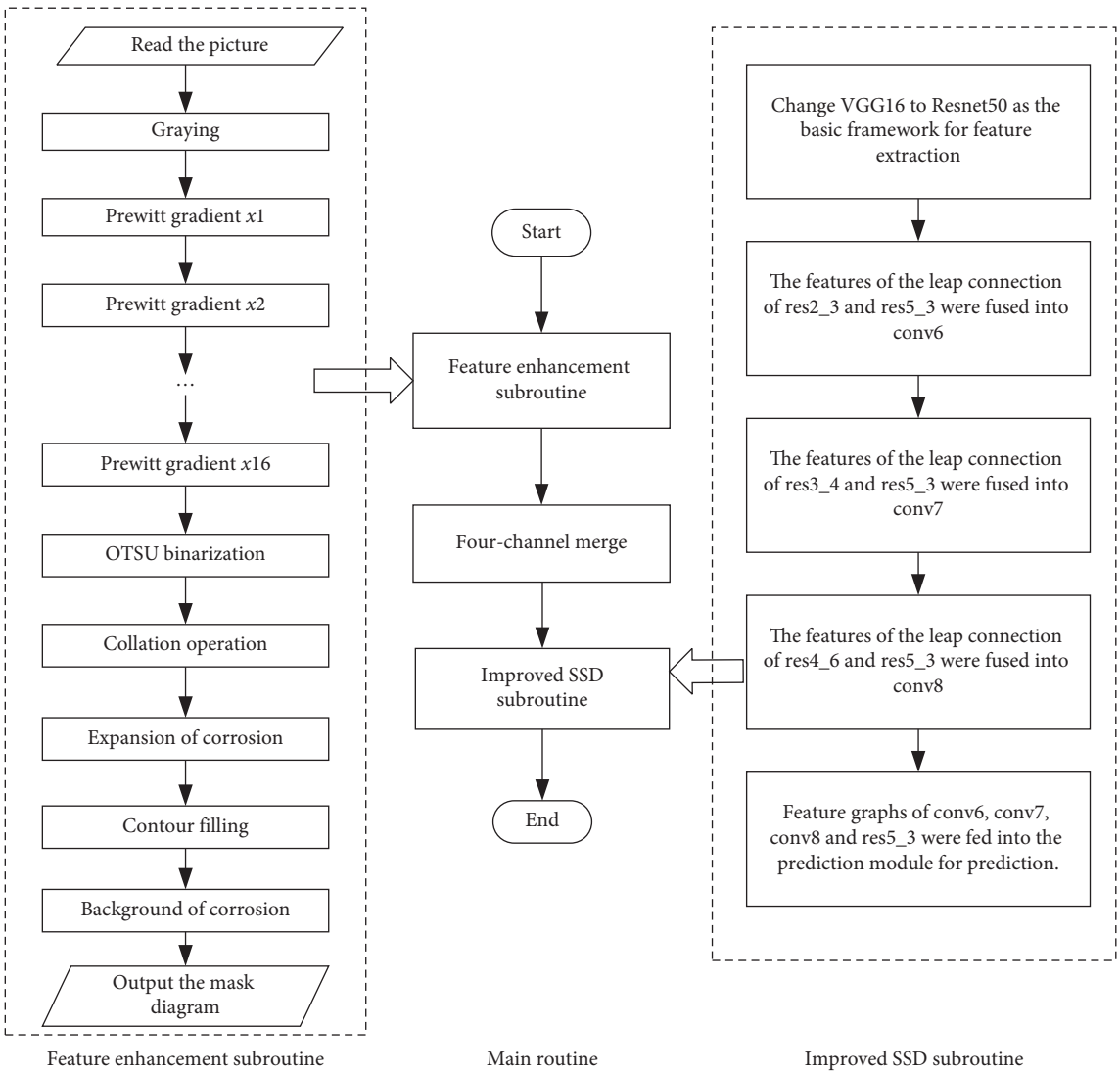


Figure 6: Underwater image detection flow chart.

Figure 7 shows the image after the Laplace operator single-direction, Prewitt operator 2-direction, Prewitt operator 4-direction, Prewitt operator 8-direction, Prewitt operator 16-direction, and Sobel operator 2-direction edge detection process. The improved algorithm model of this paper is compared with the traditional SSD algorithm loss function, the x-axis represents the epoch, and the y-axis represents the training loss. The loss function of the traditional SSD algorithm fluctuates greatly. When the training reaches 200 cycles, the performance of the two algorithms is basically stable. After the image has undergone edge detection in multiple directions, the training loss of this algorithm is significantly lower than that of the SSD algorithm.

Table 1 shows the performance indicators of sea urchin recognition with and without feature enhancement for different test models. During underwater image edge extraction through multiple directions, the performance indicators of Sobel operator 2 directions, Prewitt operator 2 directions, Prewitt operator 4 directions, Prewitt operator 8 directions, Prewitt operator 16 directions, and Laplace operator single direction are compared. The data shows that the performance of the algorithm detection target framework [32] of the Sobel operator 2-direction edge detection has been improved to 82.9 AP, which is 9.5 percentage points higher than that without the edge channel. The performance of the algorithmic target detection framework in this paper of the Prewitt operator 4-direction and 8-direction edge detection is close, and among them, Prewitt operator 8-direction is improved to 82.3% AP which is 8.9% points higher than that without the edge channel. The performance of the algorithmic target detection framework in this paper of the Prewitt operator 16-direction is improved to 83.1% AP, which is 9.7% points higher than that without the edge channel, while the edge channel of the Prewitt operator 2-direction is less effective. Using the Prewitt operator 16-direction can better extract the edge features of the sea urchin and ultimately improve the detection accuracy in the later stage.

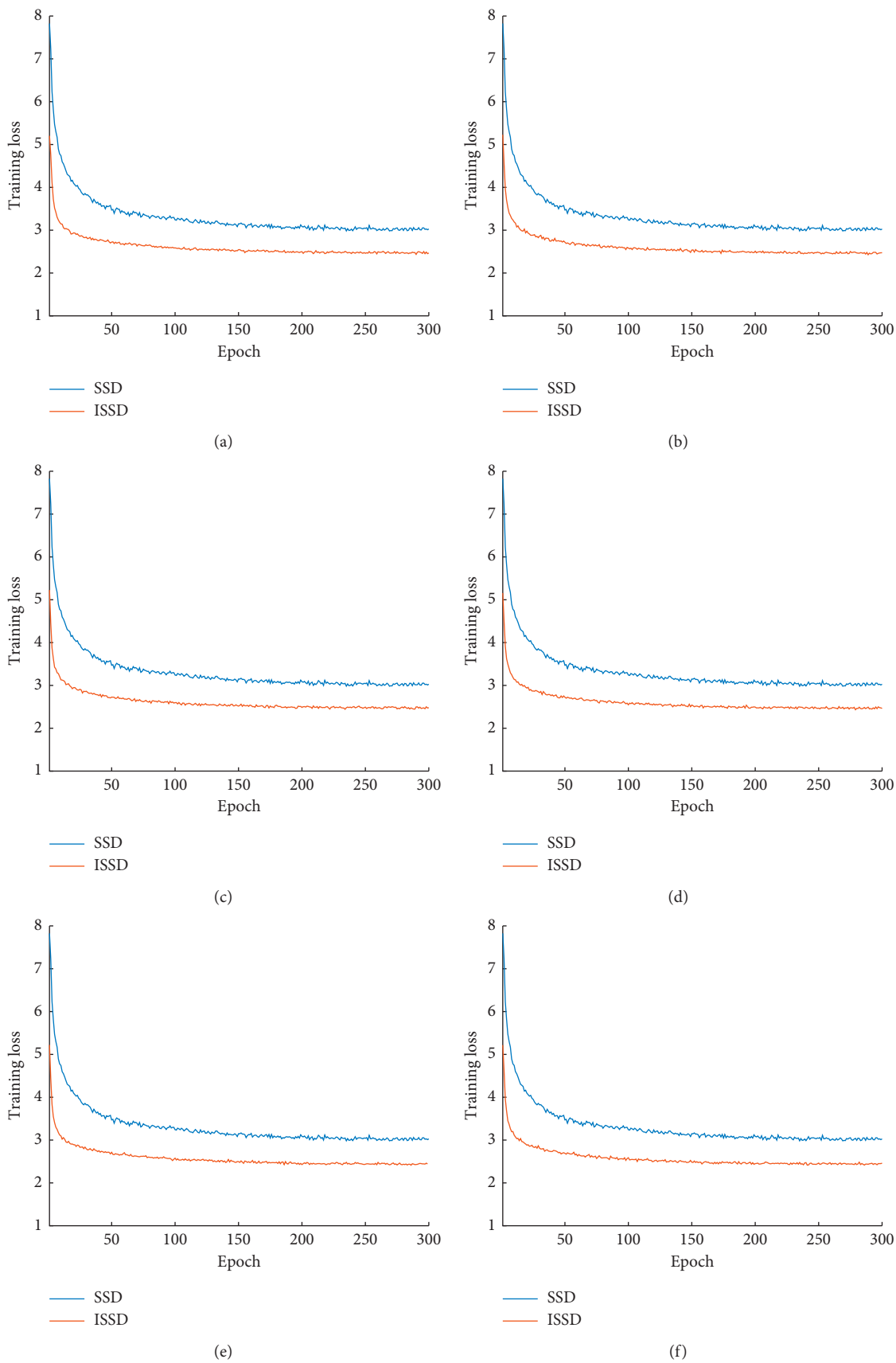


Figure 7: Comparison of loss functions of multidirection detections. (a) Loss function comparison of Laplace operator unidirectional. (b) Loss function comparison of Prewitt operator 2 directions. (c) Loss function comparison of Prewitt operator 4 directions. (d) Loss function comparison of Prewitt operator 8 directions. (e) Loss function comparison of Prewitt operator 16 directions. (f) Loss function comparison of Sobel operator 2 directions.

Table 1: Performance of sea urchin recognition under different test models with or without feature enhancement.

Model	Input size	(4th channel)	AP (%)
SSD	300 × 300	NO	73.4
ISSD	300 × 300	Sobel operator 2 directions	82.9
		Prewitt operator 2 directions	82.1
		Prewitt operator 4 directions	82.3
		Prewitt operator 8 directions	82.3
		Prewitt operator 16 directions	83.1
		Laplace operator unidirectional	82.4

4.2. Rationality Analysis of Feature Cross-Level Fusion. Analyzing the preliminary work of applying SSD to sea urchin detection, we think the main problem is that some sea urchins are small targets, and the traditional SSD algorithm has a relatively poor detection effect on small-sized objects. The feature map representation capability of shallow extraction is not strong enough. In this way, there will be misdetection and missed detection of small targets of the sea urchin. According to this, in order to improve the characteristics of the poor recognition ability of the SSD algorithm for small targets, the improved SSD algorithm draws on the idea of the residual network and uses Resnet50 instead of VGG16 as the basic framework of the network. Deepening the neural network by learning the residuals can avoid the problems of overfitting and the disappearance of the network gradient, learn more abstract texture features and semantic features, and strengthen the expression ability of features, so as to improve the ability of target classification and location. At the same time, a feature cross-level fusion method is proposed to improve the feature expression ability and strengthen the rational analysis of the semantic information feature cross-level fusion idea, mainly looking at the loss function and P-R curve during training. The convergence curve of the loss function during training is shown in Figure 8(a).

The training results of this algorithm are compared with the traditional SSD, RFBNet (Receptive Field Block Net) [33], FSSD (Feature Fusion Single-Shot Multibox Detector) [34], RefineDet (Single-Shot Refinement Neural Network for Object Detection) [35], and M2Det (Multilevel and Multiscale Detector) [36] algorithm, where the x-axis represents the period (epoch) and the y-axis represents the training loss. In the early stage of training, the RefineDet uses the Refine_multibox_loss, which is the second operation of the multibox_loss, so the convergence speed is slow; the other 5 curves use the multibox_loss as the loss function, and their convergence speeds are very fast. When the training reaches 200 cycles, the loss function of the original SSD algorithm value remains stable and no longer converges, maintaining a high loss value, and the positioning and classification losses are very large; RFBNet, FSSD, and M2Det algorithms are based on the SSD algorithm and all have a certain network architecture and feature fusion optimization to obtain better training effects; even if it reaches 200 cycles, the loss function continues to converge; the ISSD algorithm proposed in this paper adopts a method of feature cross-level fusion, which can ensure that objects of smaller scales will not have the problem of target disappearing after convolutions in deeper network layers, and it can provide improvement for the performance of small targets in the later period. It was very helpful and achieved better results during the training phase.

The AP value during training is shown in Table 2. The P-R curve is shown in Figure 8(b). The SSD algorithm has a low classification accuracy rate and recall rate index, and the highest recall rate is only 0.83. In contrast, although the highest recall rate of the RFBNet is similar to that of the SSD, its classification accuracy is higher. That is, the sea urchin in the RFBNet-based sea urchin identification system has higher confidence. The RefineDet uses the quadratic multibox_loss as the loss function, and it is similar to the Faster R-CNN detection method. In the first stage, it performs two classifications to filter a large number of samples, and then, in the second stage, it performs multiclassification to obtain the detection results, which greatly improves the accuracy rate and reduces the recall rate. The highest recall rate is only 0.80, and the lowest accuracy rate is as high as 0.72. The FSSD and M2Det algorithms have obvious advantages over the SSD and the RFBNet. The recall rate reaches 0.86, but it is difficult to achieve the desired performance. The ISSD algorithm proposed in this paper combines the advantages and disadvantages of the abovementioned three algorithms and has made a series of improvements. Finally, the performance of the algorithm has been greatly improved. Figure 8(c) shows the test results of the underwater target detection test set, and it can be seen from the figure that as the training period becomes larger, the detection accuracy of the six algorithms is constantly improving, the detection performance of the SSD algorithm fluctuates greatly, and its convergence is the fastest. In 200 training cycles, the performance of the six algorithms is basically stable. The detection accuracy of the algorithm in this paper is significantly better than that of the SSD algorithm, and the accuracy of the final test reaches 0.81. Table 2 is the AP performance index obtained by the ISSD algorithm model and the SSD, RFBNet, FSSD, RefineDet, and M2Det target detection model on the sea urchin test data set. The algorithm proposed in this paper has made a series of improvements, and finally, the performance of the algorithm has been further improved. The original SSD algorithm did not obtain the semantic information of the context due to the direct prediction of the multilayer feature map. When the confidence level is low, the recall rate is lower than that of the other three algorithms. When the confidence level is increased, the overall recall rate converges very quickly, and the confidence level of the sea urchin obtained by using the SSD algorithm is very low.

4.3. Analysis of Experimental Results. Analyzing the preliminary work of applying SSD to sea urchin detection, we believe that the sea urchin has several important features such as being black, round, and spiny, and we think that deep learning should have no pressure on the extraction of black and round features. Due to the different angles of the thorns and different convolution sizes, it may be that deep learning has the possibility of further improving the detection effect of thorns. Therefore, this paper proposes a sea

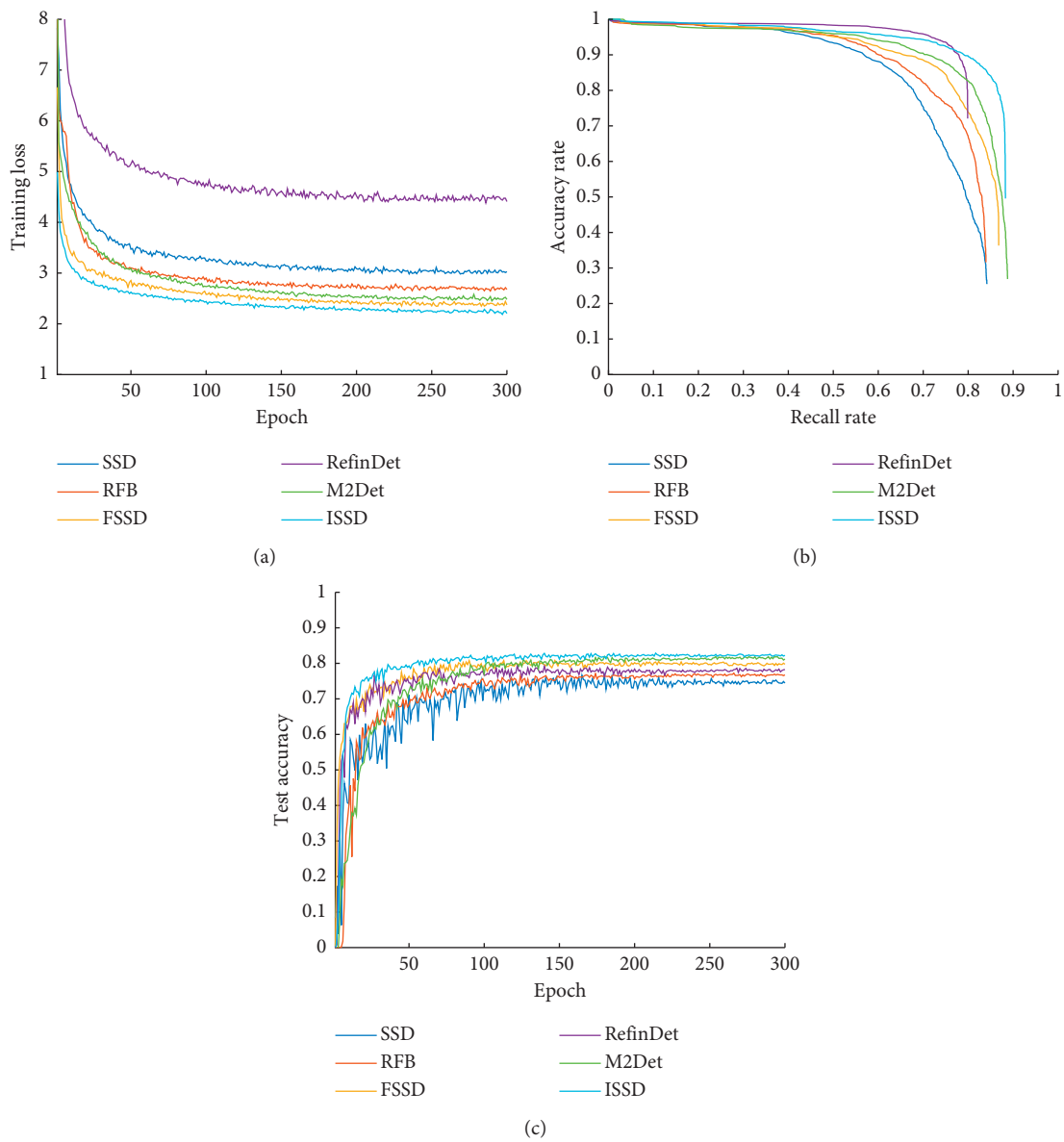


Figure 8: (a) Network loss function diagram; (b) P-R curve; and (c) accuracy curve of the test set.

Table 2: Detection and identification performance of different test models on the sea urchin test set.

Model	Input size	Model basic architecture	AP (%)
SSD	300 × 300	VGG16	73.4
RFB	300 × 300	VGG16	76.7
FSSD	300 × 300	VGG16	80.1
RefineDet	320 × 320	VGG16	78.6
M2Det	320 × 320	VGG16	80.4
ISSD	300 × 300	Resnet 50	81.0

urchin detection algorithm based on feature enhancement. According to the spiny-edge characteristics of sea urchin, a multidirectional edge detection algorithm is proposed to enhance the feature, which is taken as the 4th channel of image and the original 3 channels of underwater image together as the input for the further deep learning. At the same time, we think the main problem is that some sea urchins are small targets, and the traditional SSD algorithm has a relatively poor detection effect on small-sized objects. The feature map representation capability of shallow extraction is not strong enough. In this way, there will be misdetection and missed detection of small targets of the sea urchin. According to this, in order to improve the characteristics of the poor recognition ability of the SSD algorithm for small targets, the improved SSD algorithm draws on the idea of the residual network and uses Resnet50 instead of VGG16 as the basic framework of the network. Deepening the neural network by learning the residuals can avoid the problems of overfitting and the disappearance of the network gradient, learn more abstract texture features and semantic features, and strengthen the expression ability of features, so as to improve the ability of target classification

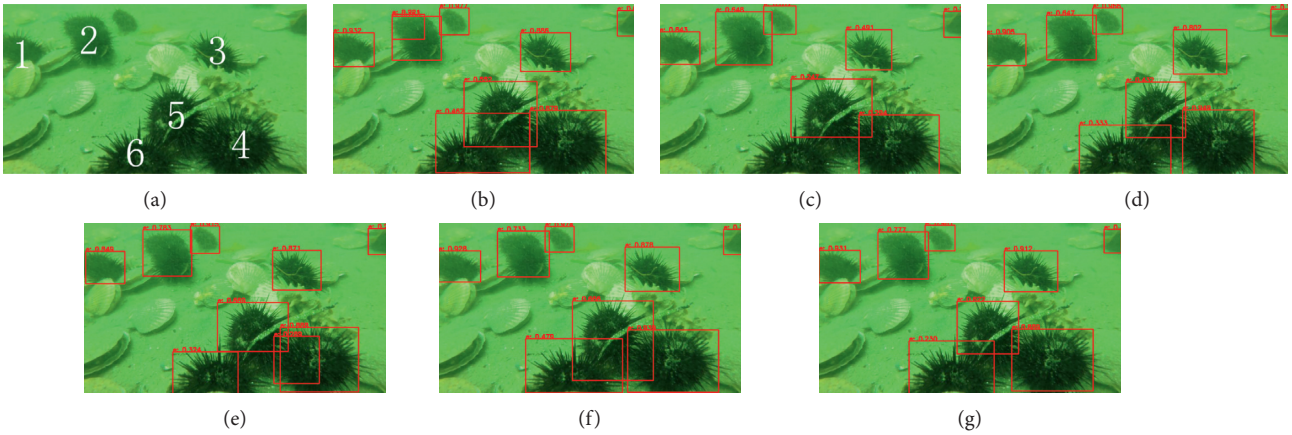


Figure 9: Multidirectional detection target detection result graph of ISSD, (a) the confidence test graph of sea urchin (Laplace operator unidirectional), (c) the confidence test graph of sea urchin (Prewitt operator 2 directions), (d) the confidence test graph of sea urchin (Prewitt operator 4 directions), (e) the confidence test graph of sea urchin (Prewitt operator 8 directions), (f) the confidence test graph of sea urchin (Prewitt operator 16 directions), and (g) the confidence test graph of sea urchin (Sobel operator 2 directions).

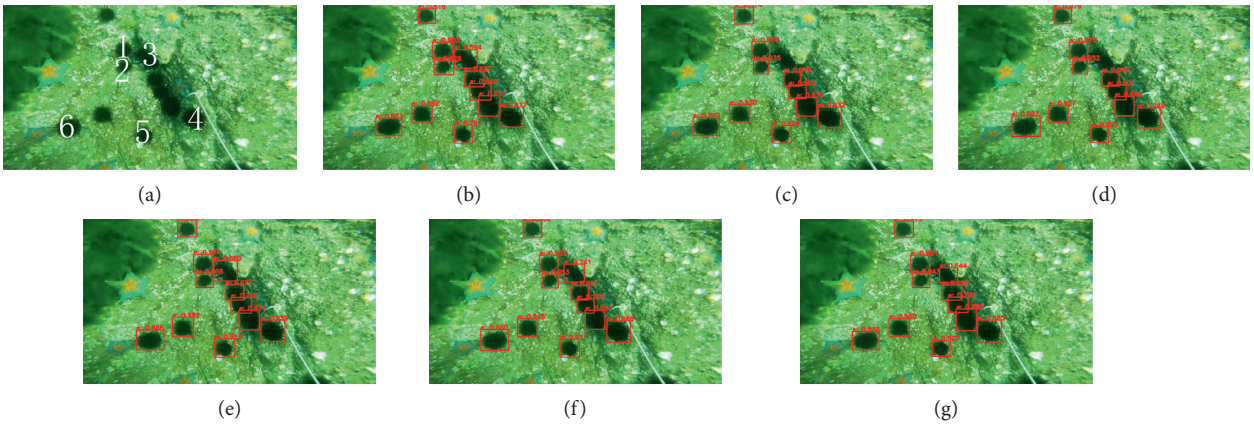


Figure 10: Multidirectional detection target detection result graph of ISSD, (a) the confidence test graph of sea urchin (Laplace operator unidirectional), (c) the confidence test graph of sea urchin (Prewitt operator 2 directions), (d) the confidence test graph of sea urchin (Prewitt operator 4 directions), (e) the confidence test graph of sea urchin (Prewitt operator 8 directions), (f) the confidence test graph of sea urchin (Prewitt operator 16 directions), and (g) the confidence test graph of sea urchin (Sobel operator 2 directions).

and location. Figures 9 and 10 are the effect diagram of the improved SSD algorithm model detection after performing the edge detection processing of the Sobel operator 2-direction, Prewitt operator 2-direction, Prewitt operator 4-direction, Prewitt operator 8-direction, Prewitt operator 16-direction, and Laplace operator single-direction. Obviously, it can be found from the figure that, after performing multidirectional edge detection on the image proposed in this paper, the algorithm of this paper has better performance for detecting small targets. In summary, the multidirectional detection algorithm proposed in this paper has better performance in sea urchin detection.

Table 3 shows the sea urchin confidence of the ISSD algorithm for multidirectional detection in Figure 9(a). Table 4 shows the sea urchin confidence of the ISSD algorithm for multidirectional detection in Figure 10(a). In the table, the values that are bold and underlined are the best and the values that are underlined are the 2nd best. It can be clearly found that the Prewitt operator 16-direction edge detection has the highest sea urchin confidence. The edge detection effect of the Prewitt operator in the 2 directions is poor, and there will be cases of missed detection.

Analyzing the preliminary work of applying SSD to sea urchin detection, we believe that the sea urchin has several important features such as being black, round, and spiny. According to the spiny-edge characteristics of sea urchin, a multidirectional edge detection algorithm is proposed to enhance the feature. The comparison of data in Tables 3 and 4 can more clearly illustrate the correctness of the sea urchin detection algorithm based on feature enhancement proposed in this paper, which is taken as the 4th channel of image and the original 3 channels of underwater image together as the input for the further deep learning. Using the Prewitt operator 16-direction can better extract the edge

Table 3: Sea urchin confidence in the ISSD algorithm for multidirectional detection in Figure 9(a).

Scene 1 sea urchin confidence	Sea urchin 1	Sea urchin 2	Sea urchin 3	Sea urchin 4	Sea urchin 5	Sea urchin 6
Laplace operator unidirectional	**0.932**	0.753	**0.886**	0.879	0.552	**0.462**
Prewitt operator 2 directions	0.843	0.648	0.491	0.294	0.247	None
Prewitt operator 4 directions	0.905	0.647	0.802	0.949	0.432	0.333
Prewitt operator 8 directions	0.849	*0.783*	0.871	0.888	**0.669**	0.324
Prewitt operator 16 directions	**0.928**	0.733	0.878	**0.939**	**0.666**	**0.479**
Sobel operator 2 directions	0.931	**0.777**	**0.912**	**0.889**	0.622	0.230

Table 4: Sea urchin confidence in the ISSD algorithm for multidirectional detection in Figure 10(a).

Scene 2 sea urchin confidence	Sea urchin 1	Sea urchin 2	Sea urchin 3	Sea urchin 4	Sea urchin 5	Sea urchin 6
Laplace operator unidirectional	0.903	0.828	0.594	0.917	0.791	**0.923**
Prewitt operator 2 directions	0.969	0.935	None	0.932	**0.930**	0.829
Prewitt operator 4 directions	0.976	0.952	None	0.949	0.901	0.887
Prewitt operator 8 directions	0.974	**0.968**	**0.680**	**0.959**	0.913	0.886
Prewitt operator 16 directions	**0.982**	**0.953**	**0.741**	**0.966**	0.917	**0.889**
Sobel operator 2 directions	**0.981**	0.943	0.644	0.865	**0.927**	0.838

features of the sea urchin and ultimately improve the detection accuracy in the later stage.

5. Conclusions

Autonomous detection and fishing by underwater robots will be the main way to obtain aquatic products in the future, and sea urchin is the main research object of aquatic product detection. The preliminary work of applying classic SSD to sea urchin detection, the existing shortcomings of inaccurate detection of small sea urchin targets, and the overall detection performance of sea urchin have room for further improvement. Therefore, this paper uses feature enhancement methods to enhance the analysis ability of deep learning on the feature learning of the thorny edge and improve the performance of detection and recognition of sea urchins. We used resnet 50 as the basic architecture for network feature extraction to replace the original VGG16 of the SSD algorithm.

ccording to the analysis of experimental data, the improvement of the classic SSD in this paper effectively improves the ability of the SSD in the sea urchin recognition task. However, the improved model still has shortcomings and will not meet the real-time requirements, mainly because the multidirectional edge detection has a large calculation amount and a long running time in image processing, so the algorithm is optimized and the calculation amount is compressed to meet the real-time requirements, and use of image enhancement and target detection for underwater robots, real-time detection, and recognition of sea urchins by underwater robots will be the focus and main direction of the next research.

Data Availability

The code used to support the findings of this study are available from the corresponding author upon request (nuistpanda@163.com, 001600@nuist.edu.cn). The data are from the open data set of the National Natural Science Foundation of China Underwater Robot Competition (http://www.cnurpc.org/a/xwjrz/2019/0808/129.html).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

All authors drafted the manuscript and read and approved the final manuscript.

Acknowledgments

The research in this article was supported by the National Natural Science Foundation of China (61773219 and 61701244) and the key special project of the National Key R&D Program (2018YFC1405703), and the authors would like to express their heartfelt thanks.

References

[1] A. Olmos and E. Trucco, "Detecting man-made objects in unconstrained subsea videos," in *Proceedings of the British Machine Vision Conference*, pp. 1–10, Cardiff University, Cardiff, UK, September 2002.

[2] M. Xia, W. A. Liu, Y. Xu, K. Wang, and X. Zhang, "Dilated residual attention network for load disaggregation," *Neural Computing and Applications*, vol. 31, no. 12, pp. 8931–8953, 2019.

[3] M. Xia, W. Liu, B. Shi, L. Weng, and L. Jia, "Cloud/snow recognition for multispectral satellite imagery based on a multidimensional deep residual network," *International Journal of Remote Sensing*, vol. 40, no. 1, pp. 156–170, 2019.

[4] J. Qian, M. Xia, and X. Yue, "Parallel knowledge acquisition algorithms for big data using MapReduce," *International Journal of Machine Learning & Cybernetics*, vol. 1, no. 2, pp. 1–15, 2015.

[5] L. Weng, X. Sun, M. Xia, J. Liu, and Y. Xu, "Portfolio trading system of digital currencies: a deep reinforcement learning with multidimensional attention gating mechanism," *Neurocomputing*, vol. 402, pp. 171–182, 2020.

[6] L. Weng, Y. Xu, M. Xia, Y. Zhang, J. Liu, and Y. Xu, "Water areas segmentation from remote sensing images using a

separable residual Segnet network," *ISPRS International Journal of Geo-Information*, vol. 9, no. 4, p. 256, 2020.

[7] Y. Bengio, "Deep learning of representations: looking forward," in *Proceedings of the International Conference on Statistical Language and Speech Processing*, Springer, Berlin, Germany, pp. 1–37, July 2013.

[8] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[9] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection,"vol. 1, pp. 886–893, in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, IEEE, San Diego, CA, USA, June 2005.

[10] T. Joachims, "Making large-scale SVM learning practical," Technical Report 1998, 28, MIT Press, Cambridge, MA, USA, 1998.

[11] D. D. Margineantu and T. G. Dietterich, "Pruning adaptive boosting," in *Proceedings of the Fourteenth International Conference on Machine Learning*, vol. 97, pp. 211–218, Nashville, TN, USA, July 1997.

[12] M. Xia, W. Song, X. Sun, J. Liu, T. Ye, and Y. Xu, "Weighted densely connected convolutional networks for reinforcement learning," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 34, no. 4, Article ID 2052001, 2020.

[13] R. Girshick, J. Donahue, T. Darrell et al., "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587, Columbus, OH, USA, June 2014.

[14] R. Girshick, "Fast R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1440–1448, Santiago, Chile, December 2015.

[15] S. Ren, K. He, R. Girshick et al., "Faster R-CNN: towards real-time object detection with region proposal networks," *Advances in Neural Information Processing Systems*, vol. 39, no. 6, pp. 1137–1149, 2015.

[16] W. Liu, D. Anguelov, D. Erhan et al., "SSD: single shot multibox detector," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Springer, Amsterdam, The Netherlands, pp. 21–37, October 2016.

[17] J. Redmon, S. Divvala, R. Girshick et al., "You only look once: unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, Las Vegas, NV, USA, June 2016.

[18] M. Bakratsas, P. Basaras, and D. Katsarosb, "Take me to SSD : a hybrid block-selection method on HDFS based on storage type," in *INNS Conference on Big Data*, pp. 111–119, Elsevier, Amsterdam, Netherlands, 2016.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, https://arxiv.org/abs/1409.1556.

[21] Y. Song, L. Zhang, S. Chen, D. Ni, B. Lei, and T. Wang, "Accurate segmentation of cervical cytoplasm and nuclei based on multiscale convolutional network and graph partitioning," *IEEE Transactions on Biomedical Engineering*, vol. 62, no. 10, pp. 2421–2433, 2015.

[22] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 315–323, Fort Lauderdale, FL, USA, April 2011.

[23] X. Li, H. Hu, L. Zhao et al., "Image recovery method combining histogram stretching for underwater imaging," *Scientific Reports*, vol. 8, no. 1, pp. 1–10, 2018.

[24] J. Liang, L. Ren, E. Qu, B. Hu, and Y. Wang, "Method for enhancing visibility of hazy images based on polarimetric imaging," *Photonics Research*, vol. 2, no. 1, pp. 38–44, 2014.

[25] C. Liu, J. Zhao, Y. Shen, Y. Zhou, X. Wang, and Y. Ouyang, "Texture filtering based physically plausible image dehazing," *The Visual Computer*, vol. 32, no. 6–8, pp. 911–920, 2016.

[26] J. Ahn, S. Yasukawa, T. Sonoda, T. Ura, and K. Ishii, "Enhancement of deep-sea floor images obtained by an underwater vehicle and its evaluation by crab recognition," *Journal of Marine Science and Technology*, vol. 22, no. 4, pp. 758–770, 2017.

[27] C. O. Ancuti, C. Ancuti, C. De Vleeschouwer, and P. Bekaert, "Color balance and fusion for underwater image enhancement," *IEEE Transactions on Image Processing*, vol. 27, no. 1, pp. 379–393, 2018.

[28] J. Y. Chiang and Y.-C. Chen, "Underwater image enhancement by wavelength compensation and dehazing," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1756–1769, 2012.

[29] A. Galdran, D. Pardo, A. Picón, and A. Alvarez-Gila, "Automatic red-channel underwater image restoration," *Journal of Visual Communication and Image Representation*, vol. 26, pp. 132–145, 2015.

[30] The National Natural Science Foundation of China Underwater Robot Competition, http://www.cnurpc.org/a/xwjrz/2019/0808/129.html.

[31] Y. Zhang, J. Zhang, P. J. Smith, M. Shafi, and P. Zhang, "Reduced complexity channel models for IMT-advanced evaluation," *EURASIP Journal on Wireless Communications and Networking*, vol. 2009, no. 1, pp. 1–13, 2009.

[32] P. Liu and Z. Li, "Task complexity: a review and conceptualization framework," *International Journal of Industrial Ergonomics*, vol. 42, no. 6, pp. 553–568, 2012.

[33] S. Liu, D. Huang, and Y. Wang, "Receptive field block net for accurate and fast object detection," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 385–400, Munich, Germany, September 2018.

[34] A. Belfodil, A. Belfodil, A. Bendimerad et al., "FSSD-a fast and efficient algorithm for subgroup set discovery," in *Proceedings of the 2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, Washington, DC, USA, October 2019.

[35] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Single-shot refinement neural network for object detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, June 2018.

[36] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen, L. Cai et al., "M2Det: a single-shot object detector based on multi-level feature pyramid network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, January 2019.

Hindawi
Complexity
Volume 2020, Article ID 2680480, 9 pages
https://doi.org/10.1155/2020/2680480

Research Article

Prediction for Chaotic Time Series-Based AE-CNN and Transfer Learning

Baogui Xin and Wei Peng

College of Economics and Management, Shandong University of Science and Technology, Qingdao 266590, China

Correspondence should be addressed to Baogui Xin; xin@sdust.edu.cn

Received 31 July 2020; Revised 28 August 2020; Accepted 10 September 2020; Published 16 September 2020

Academic Editor: Min Xia

It has been a hot and challenging topic to predict the chaotic time series in the medium-to-long term. We combine autoencoders and convolutional neural networks (AE-CNN) to capture the intrinsic certainty of chaotic time series. We utilize the transfer learning (TL) theory to improve the prediction performance in medium-to-long term. Thus, we develop a prediction scheme for chaotic time series-based AE-CNN and TL named AE-CNN-TL. Our experimental results show that the proposed AE-CNN-TL has much better prediction performance than any one of the following: AE-CNN, ARMA, and LSTM.

1. Introduction

Chaos is the unity of extrinsic randomness and intrinsic certainty, which widely exists in social fields and natural systems. However, in many cases, we cannot represent chaotic systems by establishing analytical mathematical models but by observing the time series [1]. With the development of big data acquisition and research technology, we can obtain large-scale chaotic time series from a wide range of social fields and natural systems, which highlight the necessity of research on the prediction of chaotic time series. Of course, the prediction of chaotic systems has always been a hot and challenging topic in many fields. However, the acquisition of chaotic time series data usually accompanies errors, which make the prediction of chaotic time series more difficult.

The most significant characteristic of a chaotic system is its initial sensitivity; that is, a slight change in input will produce random and unpredictable difference after long enough evolution, which determines the difficulty in medium-to-long-term prediction for a chaotic system. Similarly, the error in chaotic time series may be amplified to an unacceptable degree due to its initial sensitivity, which will lead to the failure of prediction. However, the orbit of the chaotic system also has intrinsic certainty such as self-similarity, which indicates that a chaotic system can be predicted to a certain extent, even in the medium-to-long term by using fractal geometry and other related theories. But the self-similarity and other intrinsic certainty of a chaotic system are very difficult to be measured accurately. So far, there is no commonly accepted theory that can be employed to accurately measure the intrinsic certainty such as self-similarity and realize the medium-to-long-term prediction of a chaotic system. Inspired by Xia et al. [2], we try to overcome the following difficulties to effectively predict the medium-to-long-term time series: (1) how to make full use of the intrinsic certainty of chaotic time series? It needs us to propose or borrow a universal approximation framework [3], which has a powerful representation of deep learning and can extract high-order features from input information for medium-to-long-term prediction of chaotic time series; (2) how to extract intrinsic high-order features from a chaotic time series and transfer them to its coupled time series to strengthen its prediction performance.

To solve abovementioned difficulties, we will mainly implement the following contributions:

(1) Employ a convolutional neural network (CNN) to extract high-order features of a chaotic time series for medium-to-long-term prediction. The CNN has

a strong representation learning ability, so we can use it to classify the input information according to its hierarchical structure and extract high-order features from the input information.

(2) Utilize the transfer learning (TL) theory to improve the medium-to-long-term prediction performance. We extract the intrinsic high-order features from the chaotic time series and use the transfer learning theory to overlay them with their high-order features to enhance its prediction performance.

The remainder of this article is organized as follows: in Section 2, we review the related work; in Section 3, we describe the proposed the AE-CNN-TL in detail; in Section 4, we implement some experiments and discuss the results; and in Section 5, we draw a conclusion and propose some future works.

2. Related Work

2.1. Classification of Prediction Methods for Chaotic Time Series. According to the step number of prediction, we can divide the prediction methods into the single-step prediction [4] and multistep prediction [5, 6]. In the single-step prediction, the model can be only used to predict the sequence values of the next step. In the multistep prediction, the model can be employed to predict the sequence values of the multistep, which sets higher requirements. There are two strategies of the multistep prediction: direct approach and iterative approach. Based on the direct approach, the prediction of chaotic time series can complete long-term prediction once by constructing a function mapping between input and multistep output. The idea of the direct approach is simple and easy to implement, but the direct approach needs more complex function mapping structure, which is a significant challenge. The iterative approach is to predict the time series value after only one step at a time and then take the predicted value as known and finally iterate to predict the next step. The iterative approach has low requirements for the model, but in the iterative process, the prediction error is also input into the model, which results in the error accumulation and the prediction accuracy decrease in medium-to-long term. The proposed AE-CNN-TL constructed in this article belongs to the direct approach.

2.2. Prediction of Chaotic Time Series Based on Deep Learning of Direct Strategies. Taking accurate multistep predictions with a novel recurrent neural network (RNN), Min et al. [5] constructed a new scheme to implement long-term prediction. Yi et al. [7] introduced a deep neural network-(DNN-) based method to predict the air quality in medium term. Kuo and Huang [8] presented a DNN model (EPNet), which combines CNN and LSTM to predict the electricity price in medium term. Mujeeb et al. [9] built a deep LSTM model for short- and medium-term prediction. Shen et al. [10] set up a prediction model for time series named SeriesNet, which includes an LSTM and a dilated causal convolution network. Mudassir et al. [11] established the high-performance machine learning- (ANN, SANN, SVM, and LSTM) based classification and regression models to predict Bitcoin price in short-to-medium term. Hussain et al. [12] put forward a one-dimensional convolutional neural network (1D-CNN) and extreme learning machine (ELM) to predict the one-step-ahead stream-flow in short and medium terms. Chimmula and Zhang [13] constructed an LSTM network to predict COVID-19 cases. Livieris et al. [14] designed a CNN-LSTM model to predict gold prices. Zhang and Dong [15] provided a CRNN model consisting of CNN and RNN to predict temperature changes from historical data. Li [16] proposed the KLS method fusing Kalman filter, LSTM, and SVM to forecast the carbon emission. Combining multikernel convolutional layers and convolutional LSTM layers, Asadi and Regan [17] propose a deep learning framework for spatiotemporal forecasting problems.

3. Proposal

As we know, we can employ an autoencoder (AE) to learn a representation (encoding) for a time series by training the network to ignore signal "noise," i.e., denoising. Considering CNN's advantages of shift invariant or space invariant, the CNN is most commonly applied to classify images, cluster images by similarity, and perform object recognition within scenes. In this article, we will employ CNN to extract the intrinsic high-order features including similarity from the chaotic time series to implement predictions. By combining the advantages of AE and CNN, we can use AE-CNN to predict a chaotic time series, as shown in Figure 1.

If there are a chaotic time series Y associated with X, we can extract the high-order intrinsic features from Y and utilize the transfer learning theory to overlay with the high-order intrinsic features from X to enhance the prediction performance. Thus, we can develop a prediction scheme for chaotic time series of medium-to-long-term-based AE-CNN and transfer learning, named AE-CNN-TL, as shown in Figure 2.

We assume that both time series X and Y with length T, where $X = (x_1, x_2, \ldots, x_N)^T \in \mathbb{R}^{N\times 1}$ and $Y = (y_1, y_2, \ldots, y_N)^T \in \mathbb{R}^{N\times 1}$.

From Figure 2, AE-CNN-TL consists of four processes: denoising, feature extracting, transferring learning, and forecasting. At the denoising stage, a typical AE is firstly used for data filtering so as to remove the noises from the chaotic time series. At the feature extracting stage, the denoised chaotic time series X and original time series Y would be taken as the inputs of CNNs to extract deep representations. On the transferring learning stage, the deep representations from Y can be transferred to them from X and overlapped together. At the forecasting stage, we would obtain the prediction of X through a fully connected layer.

The optimization of AE-CNN-TL is to minimize the reconstruction error of AE and the training error of the whole model. At the denoising stage, the output of AE is an approximate copy of input. Therefore, we have to minimize the reconstruction error between input and output, which could maintain the direct significance of temperature. To obtain a sound model, it is necessary to minimize the

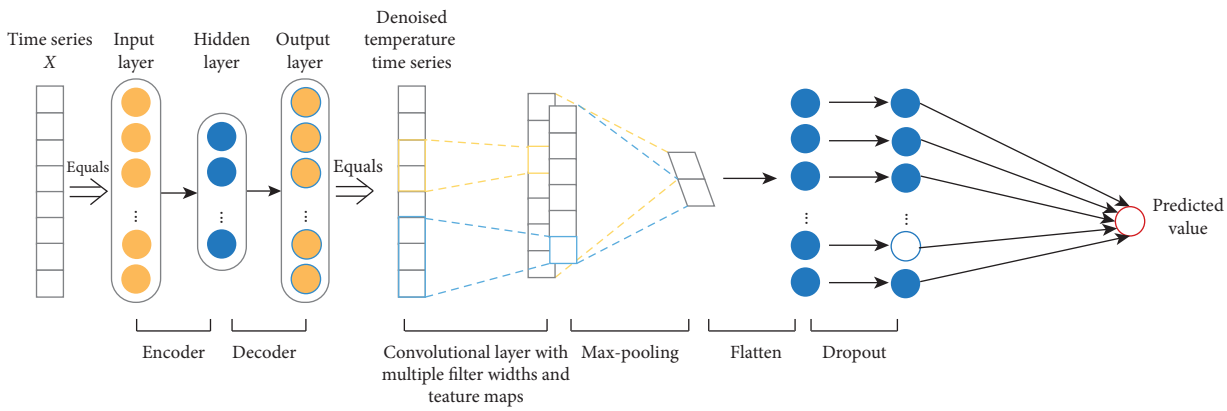


Figure 1: The architecture of AE-CNN.

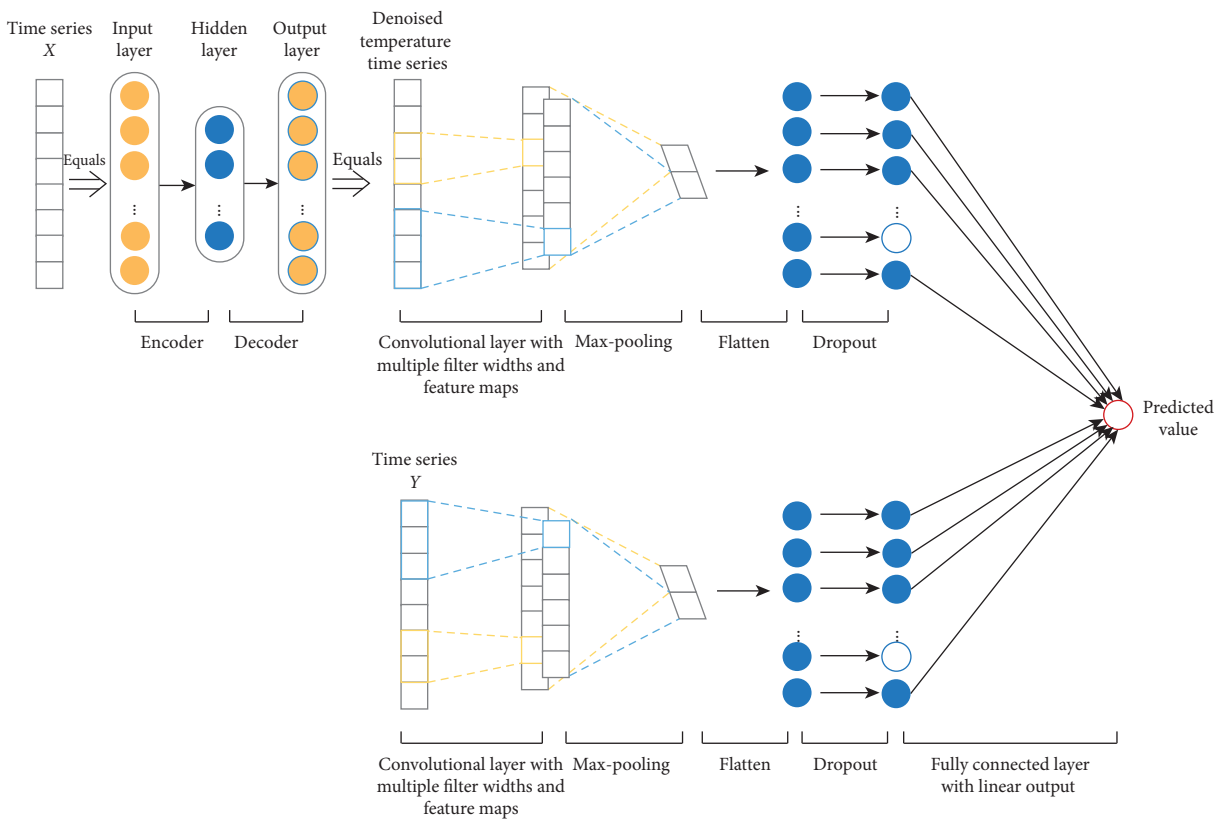


Figure 2: The architecture of AE-CNN-TL.

training error of the whole model. We formulate the objective functions as

$$e_{\mathrm{AE}} = \min \sum \left\| X - g\left(W' \times f\left(WX + b\right) + b'\right) \right\|^2, \tag{1}$$

where $f(\cdot)$ and $g(\cdot)$ are activation functions, W and W' are weights, and b and b' are biases.

It is necessary to obtain a sound model to minimize the training error of the whole model. We formulate the objective function of the whole model as

$$e_{\mathrm{AE-CNN-TL}} = \min \sum \left\| x_t - \hat{x}_t \right\|^2, \tag{2}$$

where $\hat{x}_t$ denotes the predicted value.

4. Experiments and Results

4.1. Data. In this article, we use the Beijing PM2.5 dataset (https://archive.ics.uci.edu/ml/datasets/Beijing+PM2.5+Data). The hourly data span the period between January 1, 2010, and December 31, 2014. Each series in this dataset contains 43824 data points. After removing missing values, 41757 data points were finally used in the experiment. The time series we used is PM2.5 concentration ($\mu g/m^3$) and temperature (°C). The descriptive statistics for PM2.5 concentration and temperature is shown in Table 1.

To obtain a desirable model, we divide the experimental data into three parts: 70% training dataset, 10% validation

Table 1: Descriptive statistics for PM2.5 concentration and temperature.

	PM2.5 concentration	Temperature
Mean	98.61	12.40
Median	72.00	14.00
Standard deviation	92.05	12.18
Sample variance	8,473.27	148.24
Kurtosis	7.77	1.88
Skewness	1.80	−0.14
Range	994.00	61.00
Minimum	0.00	−19.00
Maximum	994.00	42.00
Count	41,757	41,757

dataset, and 20% test dataset. The training dataset is to reach a sound model, the validation dataset is to further determine the parameters of the whole network, and the test dataset is to test the generalization ability of the model.

4.2. Evaluation Index. The root mean square error (RMSE) is a widely used evaluation index for continuous type forecasting. Generally speaking, the smaller the RMSE value is, the better its prediction performance is. The RMSE is defined as follows:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{t=1}^{N_1}\left(x_t - \hat{x}_t\right)^2}{\left|N_1\right|}}, \tag{3}$$

where N_1 denotes the length of test dataset.

4.3. 0-1 Test for Chaos in the Beijing PM2.5 Dataset. According to the 0-1 test algorithm for chaos, we sample $M = 4,176$ data points from the Beijing PM2.5 dataset after taking every 10th data point and set a random number c drawn from a uniform distribution on $(\pi/5, 4\pi/5)$ satisfying the following translation equations:

$$\begin{cases} p(n) = \sum_{j=1}^{n} X(j)\cos(jc), \\ s(n) = \sum_{i=1}^{n} X(j)\sin(jc), \end{cases} \tag{4}$$

where X denotes the time series of PM2.5 or temperature.

Then, we calculate the following test value of chaos:

$$k = \mathrm{median}\left[\frac{\mathrm{cov}(\phi,\varphi)}{\sqrt{\mathrm{cov}(\phi,\phi)\mathrm{cov}(\varphi,\varphi)}}\right], \tag{5}$$

where $\phi = (1, 2, \ldots, n_{\mathrm{cut}})$, $\varphi = (D_c(1), 2D_c(2), \ldots, D_c(n_{\mathrm{cut}}))$, and the covariance is written with vectors ϕ and φ of length r as follows:

$$\mathrm{cov}\,(\phi,\varphi) = \frac{1}{r}\sum_{j=1}^{r}\left[\phi(j) - \frac{1}{r}\sum_{j=1}^{r}\phi(k)\right]\left[\varphi(j) - \frac{1}{r}\sum_{j=1}^{r}\varphi(j)\right], \tag{6}$$

$$\begin{aligned} D_c(n) = \lim_{M\longrightarrow\infty}\frac{1}{M}\cdot\sum_{j=1}^{M}\Big[&\left(p_c(j+n) - p_c(j)\right)^2 \\ &+ \left(s_c(j+n) - s_c(j)\right)^2\Big] \\ &- \frac{1-\cos nc}{1-\cos c}\left[\lim_{N\longrightarrow\infty}\frac{1}{M}\sum_{k=1}^{M}X(j)\right]^2. \end{aligned} \tag{7}$$

Figure 3 shows plots of the test value k vs. random number c for the time series of temperature. $k \approx 1$ implies that there is a chaotic attractor in the time series of temperature.

Figure 4 shows Brownian-like trajectories and denotes there is also a chaotic attractor in the time series of temperature. That is to say, the time series of temperature is chaotic.

Similar to Figure 3, Figure 5 also shows plots of the test value k vs. random number c for the PM2.5 concentration time series. $k \approx 1$ also implies there is a chaotic attractor in the time series of PM2.5 concentration.

Similar to Figure 4, Figure 6 also shows Brownian-like trajectories and denotes there is also a chaotic attractor in the time series of PM2.5 concentration. That is to say, the time series of PM2.5 concentration is chaotic.

Figure 7 demonstrates the phase portrait of temperature vs. PM2.5 concentration, in which Brownian-like (unbounded) trajectories imply there exists a chaotic attractor.

In short, Figures 3–7 prove that there exist chaotic attractors in the Beijing PM2.5 dataset.

4.4. Experimental Results. In this part, we test the prediction performance of the proposed AE-CNN-TL for chaotic time series.

To compare the prediction performance of AE-CNN and AE-CNN-TL, we firstly employ AE-CNN to predict PM2.5 concentration without TL from temperature, as shown in Figure 8. Then, we utilize AE-CNN-TL to predict PM2.5 concentration, as shown in Figure 9. In the same way, we firstly employ AE-CNN to predict temperature without TL from PM2.5 concentration, as shown in Figure 10. Then, we utilize AE-CNN-TL to predict temperature, as shown in Figure 11. Owing to the continuous value prediction, we employ RMSE to measure the prediction performance. The smaller the RMSE value is, the better the prediction

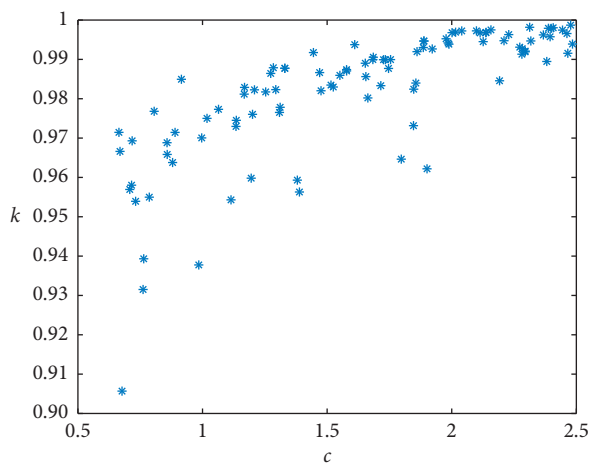


Figure 3: Plots of K vs. c for temperature.

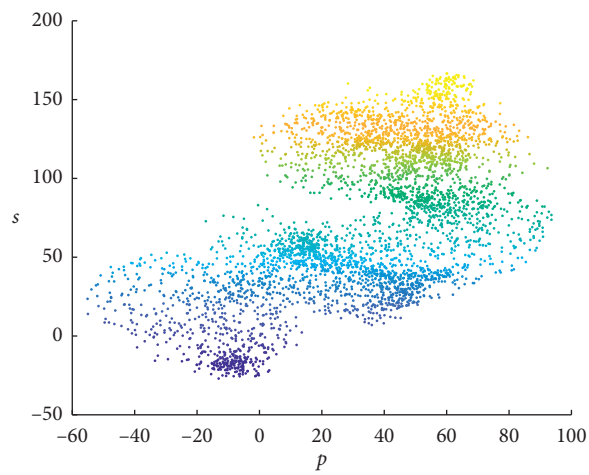


Figure 4: Plots in the new coordinate p-s plane for temperature.

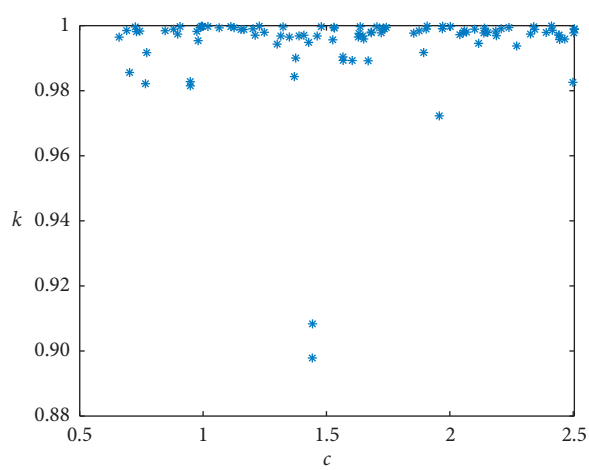


Figure 5: K vs. c for PM2.5 concentration.

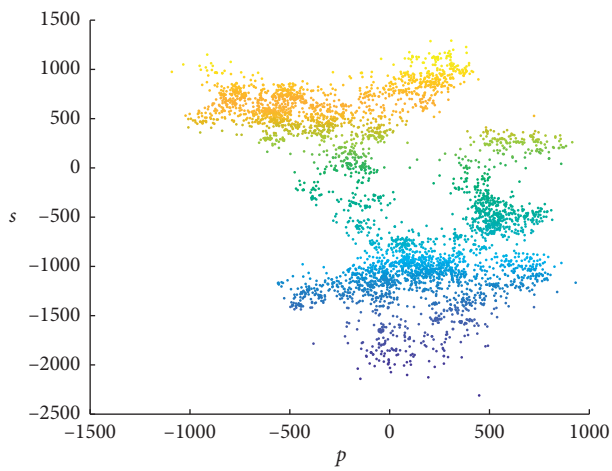


Figure 6: Phase in p-s plane for PM2.5 concentration.

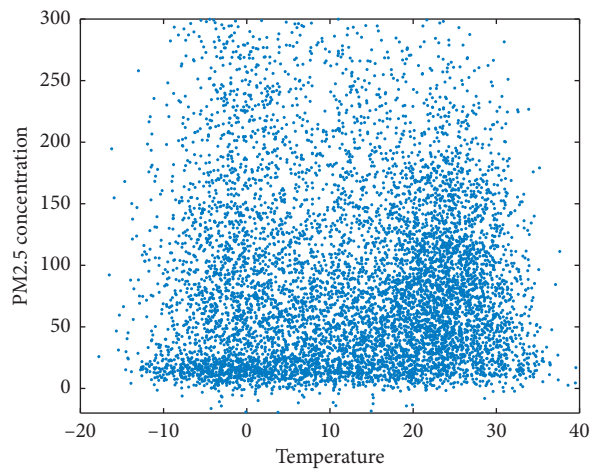


Figure 7: Phase portrait of temperature vs. PM2.5 concentration.

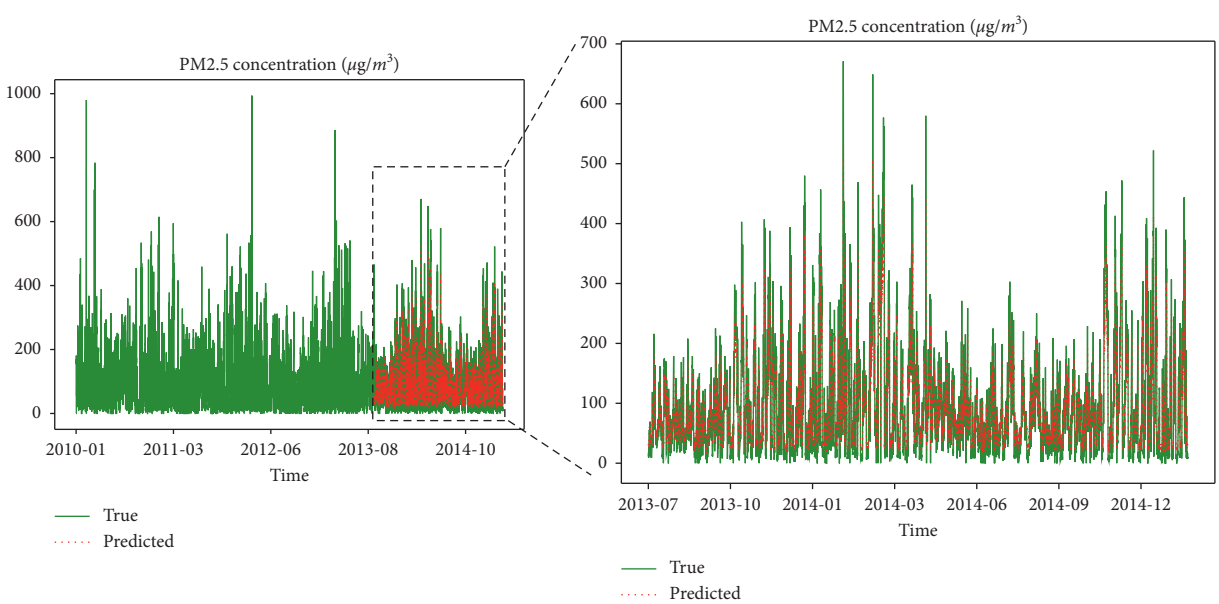


Figure 8: PM2.5 concentration prediction without TL from temperature.

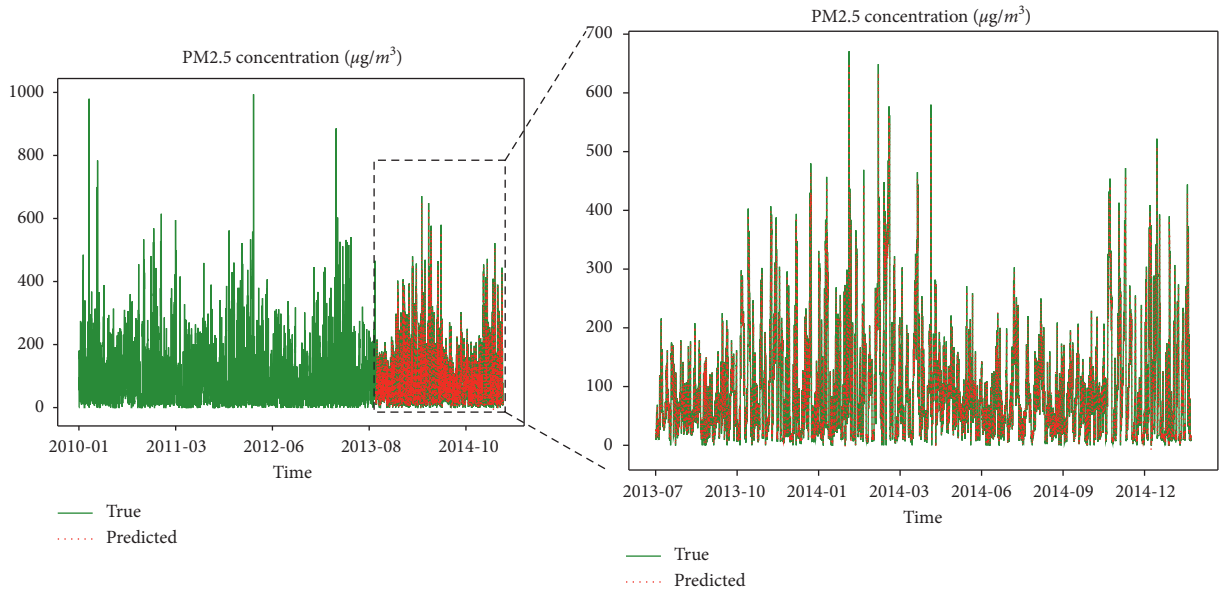


Figure 9: PM2.5 concentration prediction with TL from temperature.

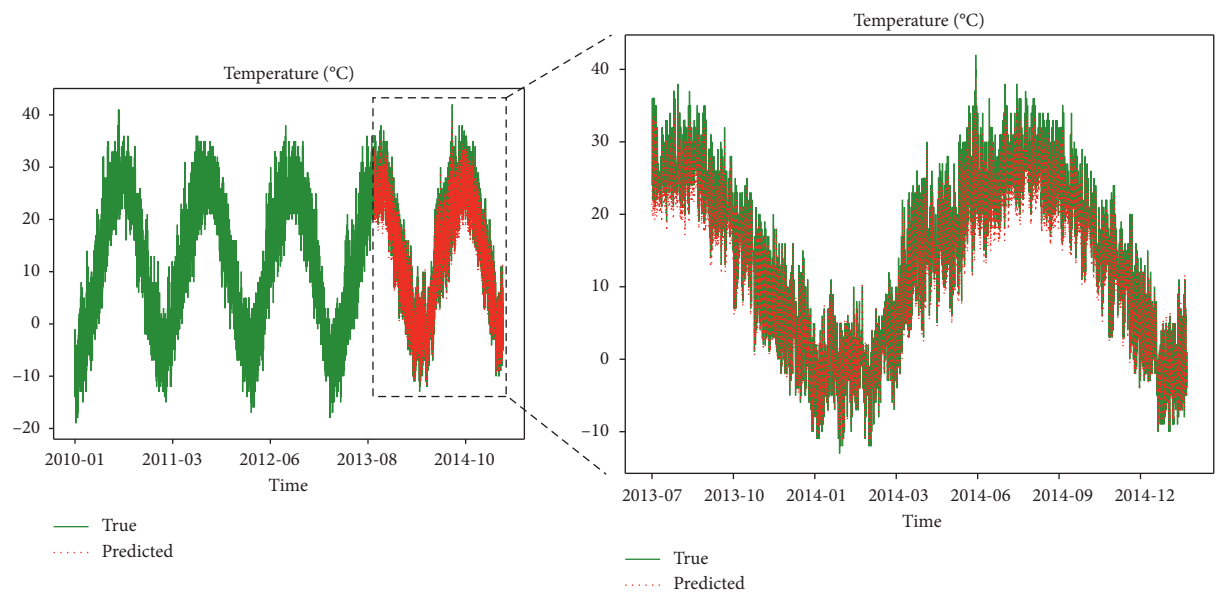


Figure 10: Temperature prediction without TL from PM2.5 concentration.

performance is. And these corresponding RMSEs of prediction are shown in Table 2.

From Table 2, we can catch that the prediction accuracy of PM2.5 concentration is enhanced to a large extent with taking TL from temperature. And the prediction accuracy of temperature is also improved significantly with taking TL from PM2.5 concentration. Therefore, we can conclude that AE-CNN-TL is much better in the prediction performance than any one of the following: AE-CNN, ARMA, and LSTM.

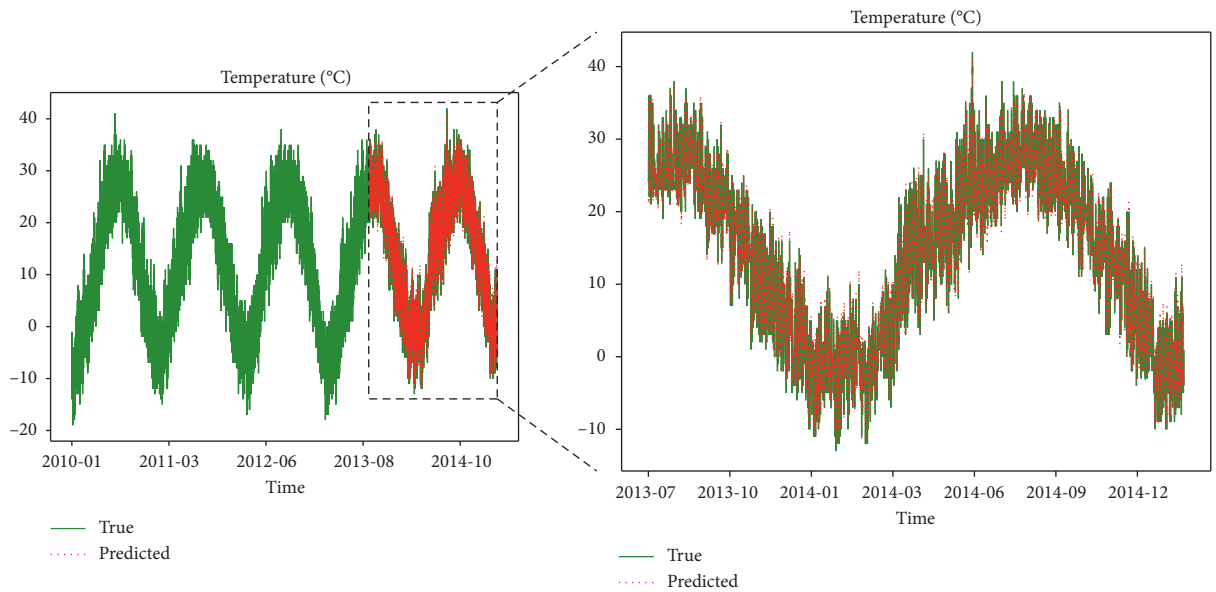


Figure 11: Temperature prediction with TL from PM2.5 concentration.

Table 2: RMSEs of prediction.

Prediction task	Prediction approach	RMSE
PM2.5 concentration prediction	ARMA	79.21
	LSTM	43.50
	AE-CNN	89.55
	AE-CNN-TL	21.53
Temperature prediction	ARMA	72.47
	LSTM	37.68
	AE-CNN	12.39
	AE-CNN-TL	1.61

5. Conclusions

In this article, we develop a prediction scheme for chaotic time series of medium-to-long-term-based AE-CNN and TL, named AE-CNN-TL, which has much better prediction performance than any one of the following: AE-CNN, ARMA, and LSTM. This prediction scheme can be used in chaotic time series of medium-to-long term of many natural fields and social systems in the real world.

In fact, AE-CNN-TL can also be used to reveal the dynamic link between PM2.5 concentration and temperature. What is more, AE-CNN-TL can be used to explore the Granger causality [18] between PM2.5 concentration and predict PM2.5 concentration and temperature with a good performance. By means of experiments, we find a bidirectional Granger causality between PM2.5 concentration and temperature, which is consist with previous researches.

We know that the Beijing PM2.5 dataset originates from the nature field. In fact, we can try to employ the proposed AE-CNN-TL to predict a chaotic time series of medium-to-long term which comes from some artificial systems, such as the game system [19–21] and the financial system [22].

Data Availability

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation, to any qualified researcher.

Conflicts of Interest

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

Acknowledgments

This work was supported by the Natural Science Foundation of Shandong Province (Grant no. ZR2016FM26) and the National Social Science Foundation of China (Grant no. 16FJY008).

References

[1] J. Ma, Y. Chen, and B. Xin, "Study on prediction methods for dynamic systems of nonlinear chaotic time series," *Applied Mathematics and Mechanics*, vol. 25, pp. 605–611, 2004.

[2] M. Xia, W. A. Liu, K. Wang, X. Zhang, and Y. Xu, "Non-intrusive load disaggregation based on deep dilated residual network," *Electric Power Systems Research*, vol. 170, pp. 277–285, 2019.

[3] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[4] W. Ji and K. C. Chee, "Prediction of hourly solar radiation using a novel hybrid model of ARMA and TDNN," *Solar Energy*, vol. 85, no. 5, pp. 808–817, 2011.

[5] H. Min, X. Jianhui, X. Shiguo, and Y. Fu-Liang, "Prediction of chaotic time series based on the recurrent predictor neural network," *IEEE Transactions on Signal Processing*, vol. 52, pp. 3409–3416, 2004.

[6] X. Wu, Y. Wang, J. Mao, Z. Du, and C. Li, "Multi-step prediction of time series with random missing data," *Applied Mathematical Modelling*, vol. 38, no. 14, pp. 3512–3522, 2014.

[7] X. Yi, J. Zhang, Z. Wang, T. Li, and Y. Zheng, "Deep distributed fusion network for air quality prediction," in *Proceedings of Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 965–973, London, UK, 2018.

[8] P.-H. Kuo and C.-J. Huang, "An electricity price forecasting model by hybrid structured deep neural networks," *Sustainability*, vol. 10, no. 4, p. 1280, 2018.

[9] S. Mujeeb, N. Javaid, M. Ilahi, Z. Wadud, F. Ishmanov, and M. Afzal, "Deep long short-term memory: a new price and load forecasting scheme for big data in smart cities," *Sustainability*, vol. 11, no. 4, p. 987, 2019.

[10] Z. Shen, Y. Zhang, J. Lu, J. Xu, and G. Xiao, "A novel time series forecasting model with deep learning," *Neurocomputing*, vol. 396, pp. 302–313, 2020.

[11] M. Mudassir, S. Bennbaia, D. Unal, and M. Hammoudeh, "Time-series forecasting of bitcoin prices using high-dimensional features: a machine learning approach," *Neural Computing and Applications*, 2020.

[12] D. Hussain, T. Hussain, A. A. Khan, S. A. A. Naqvi, and A. Jamil, "A deep learning approach for hydrological time-series prediction: a case study of gilgit river basin," *Earth Science Informatics*, vol. 13, no. 3, pp. 915–927, 2020.

[13] V. K. R. Chimmula and L. Zhang, "Time series forecasting of covid-19 transmission in Canada using LSTM networks," *Chaos, Solitons & Fractals*, vol. 135, Article ID 109864, 2020.

[14] I. E. Livieris, E. Pintelas, and P. Pintelas, "A CNN-LSTM model for gold price time-series forecasting," *Neural Computing and Applications*, 2020.

[15] Z. Zhang and Y. Dong, "Temperature forecasting via convolutional recurrent neural networks based on time-series data," *Complexity*, vol. 2020, Article ID 3536572, 8 pages, 2020.

[16] Y. Li, "Forecasting Chinese carbon emissions based on a novel time series prediction method," *Energy Science & Engineering*, vol. 8, no. 7, pp. 2274–2285, 2020.

[17] R. Asadi and A. C. Regan, "A spatio-temporal decomposition based deep neural network for time series forecasting," *Applied Soft Computing*, vol. 87, Article ID 105963, 2020.

[18] W. Peng, "DLI: a deep learning-based granger causality inference," *Complexity*, vol. 2020, Article ID 5960171, 8 pages, 2020.

[19] B. Xin, F. Cao, W. Peng, and A. A. Elsadany, "A bertrand duopoly game with long-memory effects," *Complexity*, vol. 2020, Article ID 2924169, 7 pages, 2020.

[20] B. Xin, W. Peng, and Y. Kwon, "A discrete fractional-order cournot duopoly game," *Physica A: Statistical Mechanics and its Applications*, vol. 558, Article ID 124993, 2020.

[21] B. Xin and M. Sun, "A differential oligopoly game for optimal production planning and water savings," *European Journal of Operational Research*, vol. 269, no. 1, pp. 206–217, 2018.

[22] B. Xin, W. Peng, Y. Kwon, and Y. Liu, "Modeling, discretization, and hyperchaos detection of conformable derivative approach to a financial system with market confidence and ethics risk," *Advances in Difference Equations*, vol. 2019, no. 1, p. 138, 2019.