Big Data Research 26 (2021) 100269

Contents lists available at ScienceDirect

Big Data Research

www.elsevier.com/locate/bdr

FDN-learning: Urban $PM_{2.5}$-concentration Spatial Correlation Prediction Model Based on Fusion Deep Neural Network ☆

Guojian Zou [a,b,c,1], Bo Zhang [a,d,e,*], Ruihan Yong [a,2], Dongming Qin [f], Qin Zhao [a]

[a] College of Information, Mechanical, and Electrical Engineering, Shanghai Normal University, Shanghai 200234, PR China
[b] The Key Laboratory of Road and Traffic Engineering, Ministry of Education, Tongji University, Shanghai 201804, PR China
[c] College of Transportation Engineering, Tongji University, Shanghai 201804, PR China
[d] Institute of Artificial Intelligence on Education, Shanghai Normal University, Shanghai 200234, PR China
[e] Shanghai Engineering Research Center of Intelligent Education and Bigdata, Shanghai Normal University, Shanghai 200234, PR China
[f] College of Electronics and Information Engineering, Tongji University, Shanghai 201804, PR China

ARTICLE INFO

Article history:
Received 10 July 2019
Received in revised form 15 April 2021
Accepted 21 August 2021
Available online 30 August 2021

Keywords:
Fusion deep neural network
Gaussian function
LSTM
$PM_{2.5}$-concentration prediction
Stacked anti-autoencoder

ABSTRACT

The problem of increasing air pollution poses a challenge to smart city development, as spatial air pollution correlation exists among adjacent cities. However, it is difficult to predict the degree of air pollution of a location by exploiting massive air pollution datasets incorporating data on spatially related locations. Construction of a spatial correlation prediction model for air pollution is therefore required for air pollution big-data mining. In this paper, we propose an air pollution-concentration spatial correlation prediction model based on a fusion deep neural network called FDN-learning. Three models are combined: a stacked anti-autoencoder network, Gaussian function model, and long short-term memory network (LSTM). The FDN-learning model is composed of three layers for feature expansion, intermediate processing, and data prediction. In the first layer, we employ a stacked anti-autoencoder model to learn the source-data spatial features through a feature expansion hidden layer; this can enrich the feature vector and mine more information for further prediction. In the second layer, the Gaussian function evaluates effective weights for the outputs of the stacked anti-autoencoder models in the preceding layer; the spatial correction effects are therefore incorporated in this layer. Finally, the LSTM model in the data prediction layer learns the air pollution-concentration temporal features. A fine-tuning method based on stochastic gradient descent is applied to the FDN-learning model for improved performance. Empirical results are used to verify the feasibility and effectiveness of our proposed model based on a real-world air pollution dataset.

1. Introduction

In recent years, the increasingly serious problem of air pollution has caused widespread concerns around the world [13]. The prediction of air pollutant concentration, or simply air pollutant prediction, plays a significant role in air pollution prevention and environment management [28], therefore it has received great attention recently in the research community and has been recognized as a key challenge in environment management research [57].

Air pollution has traditionally been seen as a time series processing problem; it could be predicted based on historical related data, e.g., meteorological factors (e.g. humidity wind direction) and other pollutant factors (e.g. PM_{10} and SO_2). Much work has proven

☆ This document is the results of the research project funded by the National Natural Science Foundation of China (61572326, 61802258, 61702333), Natural Science Foundation of Shanghai (18ZR1428300), and the Shanghai Committee of Science and Technology (17070502800).

* Corresponding author at: College of Information, Mechanical, and Electrical Engineering, Shanghai Normal University, Shanghai 200234, PR China.
E-mail addresses: 2010768@tongji.edu.cn (G. Zou), zhangbo@shnu.edu.cn (B. Zhang), yongruihan@163.com (R. Yong), qindm@3clear.com (D. Qin), q_zhao@shnu.edu.cn (Q. Zhao).

[1] G. Zou was with the College of Information, Mechanical and Electrical Engineering, Shanghai Normal University, Shanghai 200234, PR China, and now is with The Key Laboratory of Road and Traffic Engineering, Ministry of Education, Tongji University, Shanghai 201804, PR China.

[2] This author contributed equally to this work and should be considered co-first author.

https://doi.org/10.1016/j.bdr.2021.100269
2214-5796/

that there is a complex interaction between these factors for air pollutants [5,12,60,38,36]. Therefore, the features of these complex relationships must be extracted and learned for further air pollutant prediction. In addition, air pollution is a regional diffusion problem with a spatial dimension, i.e., there is a spatial correlation between air pollution impacts in adjacent cities [29,1,30,58]. Nevertheless, most existing works have focused on air pollutant prediction in an individual city with its historical data rather than a spatial correlation prediction in an adjacent region. In this study, we propose a new model to predict the air pollutant concentration of a target city based on its neighboring cities' related historical data (i.e., meteorological data and air pollution data) by taking into consideration the spatiotemporal features in the prediction.

In many existing works on the prediction of air pollution concentration [60,38,9,42,37,39,6,45,8,51,40,55], a numerical prediction method is widely used, which can employ historical air pollutants to realize the prediction analysis of the future state of pollution. Most numerical prediction models include a deterministic model based on hypothesis theory and prior knowledge; an empirical model that only considers input and output as an independent process; a mathematical statistics model; or a traditional machine learning model with small sample data [9,42,37,39]. The main advantages of these models are low computational complexity, fast calculation speed, and ease of implementation. However, by dealing with the massive amount of spatiotemporal data from multi-city sites for a spatial correlation air pollutant concentration prediction, traditional numerical analysis models have encountered three problems: (1) traditional methods rely excessively on historical experience and regularity. Predictions are based on the regularity determined from the historical data, and the pollution regularity is regarded as consistent; and (2) sometimes there are insufficient original data features, in which case the data features cannot be learned adequately and the deep relationships within the data are neglected; and (3) the pollutant concentration of the target city at a definite time is not an independent variable. These problems have led to most traditional air pollutant prediction models performing poorly.

Air pollutants are part of a dynamic process in the spatial dimension, susceptible to meteorological factors such as humidity, temperature, and wind direction, and pollutants are prone to propagation and spread between cities [29,1,30,58]. The concentration of air pollutants is continuous in the time dimension, and it will change dynamically with time [26]. Existing air pollutant and meteorological data have the problem of a low feature dimension, and traditional methods have difficulty learning the potential correlation features between multiple data points. Therefore, to improve the accuracy of air-pollutant concentration prediction in a target city, we should jointly consider the spatiotemporal distribution features of pollutant and meteorological data in several surrounding cities, and deeply extract the distribution features of pollutant and meteorological data.

Deep learning models have proved to perform better at spatiotemporal prediction, especially in the fields of image recognition, natural language processing (NLP), and historical data-based prediction (including air pollutant concentration prediction) [5,39,26,17,15,7,19,23,52]. In particular, experimental results for $PM_{2.5}$ prediction have proved that the deep network structure of neural network models performs better than traditional pollutant prediction methods and machine learning algorithms, because the deep features of spatial dimensions and time-dependent features can be learned more accurately [43,32,16,4,58,59,27,24,47,53]. Based on the above concerns, we introduce two kinds of artificial neural networks and a data-weighting mechanism, i.e., a stacked anti-autoencoder, Gaussian function, and long short-term memory network (LSTM), to construct our prediction model. The rationales are as follows:

(1) The proven ability of the stacked autoencoder to extract the distribution features of data through unsupervised learning has attracted much attention from researchers [56,46,54]. To deeply extract the hidden distribution features and distribution law in low-dimensional pollutant and meteorological data, we propose a new type of network structure, the stacked anti-autoencoder, which expands the data dimension while maintaining the advantages of the original autoencoder.
(2) Gaussian functions are widely used in distance-related weighting problems and have achieved good results in experiments [14,31,3,10]. In the spatial dimension, the diffusion of pollutants is also affected by the location of the city and the distance between cities [29,1,30,58,22]. The Gaussian function can use the latitude and longitude of the city, that is, weight city pollutant and meteorological data through two important factors, the city location and the distance between cities. The advantages of the Gaussian function are [14,31,3,10]: first, it fully considers the two important factors of city location and distance between cities; second, it has low computational complexity and has a strong theoretical background as support; third, it avoids relying on a large number of training samples to learn network parameters, and then implement a weighting mechanism. Therefore, we utilize Gaussian function as the preferred method to calculate the influence of surrounding cities on the concentration of air pollutants in the target city.
(3) In the time dimension, LSTM is a type of recurrent neural network (RNN) [44] that has been proposed to predict future outputs using past inputs. LSTM has been shown to be well-suited for prediction based on time series data, with better performance than RNNs with exploding- and vanishing-gradient problems [2,21,41]. Therefore, we use LSTM to predict future air pollutant concentrations by learning features contained in past air pollutant concentration time series data.

This study aims to construct a predictive model that can deeply exploit the hidden distribution features of pollutant and meteorological data through feature expansion methods and comprehensively consider the influence of surrounding cities (i.e., the spatial correlation) on pollutant concentration prediction for a target city. The main contributions of this paper are as follows:

1) Extraction of the data-expansion features is the primary purpose of the prediction system. This step is performed using anti-autoencoders. The aim is to extract the true features from the input data and to mine hidden distribution features between data. Then the model enters the feature maps produced by the stacked anti-autoencoders into the Gaussian function layer;
2) The Gaussian function layer calculates the Gaussian function value of the geographic locations of the surrounding cities and the target city, and the output of each stacked anti-autoencoder is weighted and summed to produce a one-dimensional spatial correlation feature vector. This layer fully considers the influence of the location information on the spatial correlation feature extraction, and it is an important part of the spatial correlation prediction task;
3) The memory function of the LSTM network can account for the data dependence on the temporal dimension. Thus the accuracy of the model's time series prediction results is improved; and
4) Experiments on the dataset show that our framework (FDN-Learning) achieves better results than state-of-the-art methods.

2. Related work

According to the characteristics of the prediction methods used in related studies, air pollutant concentration prediction can be

fundamentally divided into two major research methods: deterministic and statistical approaches [5,34,33,25].

Deterministic approaches can be applied to a limited set of historical data. However, meteorological principles and statistical approaches are needed to simulate the process of real-time emission, diffusion, transformation, and removal of pollutants based on atmospheric physics and chemical reactions. The model structure based on the deterministic approaches is predefined based on certain theoretical assumptions and prior knowledge. There are several commonly used methods for air pollutant concentration prediction based on deterministic approaches: comprehensive air quality model with extensions (CAMx), the WRFChem model, nested air quality prediction modeling system (NAQPMS), and the community multiscale air quality (CMAQ) model [60,38,6,45].

Statistical approaches can avoid the use of complex theoretical models. Compared with deterministic approaches, they can determine the correlations among complex pollutant concentration data and thus show better predictive performance. Based on the statistics, the two branches of approaches can be extended into traditional machine learning methods, and new deep learning methods. Traditional machine learning methods include a support vector machine (SVM), multi-label classifier based on Bayesian networks, the support vector regression (SVR) method, hidden Markov model (HMM), the multiple linear regression (MLR) method, and other methods [39,7,8,51,40,55]. In recent years, deep learning technology has excelled in dealing with regression problems, and various artificial neural networks (ANNs) based on backpropagation (BP) algorithms have been used to improve air pollution concentration prediction performance. Typical network models for predicting air pollution include the multilayer perceptron (MLP), recurrent neural network (RNN), LSTM neural network, the latest proposed deep CNN-LSTM model, graph convolutional neural network, and attention-based neural networks, etc. [13,28,57,5,12,36,26,18,34,33,11,20,43,32,16,4,58,59,27,24,47,53,49].

Because the emission, diffusion, conversion, and removal of air pollutants are a dynamic process over time, the RNN characteristic is that it can process the time series data prediction problem and easily extract temporal features of pollutant concentrations. In recent years, LSTM has been successfully applied to many studies on time series prediction problems, such as forecasting the daily maximum price of stocks, machine translation, and speech recognition [19,23,52]. The LSTM network has also been applied to air pollutant concentration prediction. However, these prediction methods based on the LSTM network use a city's multi-site historical monitoring data to predict the concentration of pollutants, and do not make full use of pollutant and meteorological data from multiple cities [26,18,34]. The new pollutant concentration prediction model, CNN-LSTM, combines the pollution data of multiple sites to realize the extraction of spatial and temporal features [36,49]. However, the CNN-LSTM has two key problems. First of all, it is difficult for a CNN to extract the hidden spatial features of pollutant and meteorological data in depth, which can easily lead to loss of feature information and degradation of the model. Second, the impact of pollutants from surrounding cities on the target city is not considered, which leads to insufficient accuracy of pollutant concentration prediction.

In recent studies on pollutant concentration prediction, methods based on graphs and attention have been used to extract the spatiotemporal features of multi-site pollutant data, and the accuracy of prediction has been improved to a certain extent [59,35]. In addition, progress has been made in spatiotemporal feature extraction based on attention and ConvLSTM. In references [48,27,47], attention-based ConvLSTM is used in recognition, traffic flow prediction and pollutant concentration prediction tasks, and has achieved satisfactory performance. In the task of pollutant concentration prediction, the prediction model based on Att-ConvLSTM uses spatiotemporal attention and ConvLSTM methods to weight the input data and extract spatiotemporal features, and the accuracy of the final prediction results has been greatly improved [24]. However, the current pollutant concentration prediction model based on Att-ConvLSTM has encountered the following challenges: First, it is difficult to deeply extract the hidden distribution features of pollutant and meteorological data. Second, it lacks consideration of the impact of multiple pollutants and meteorological factors on the prediction results; Third, the ConvLSTM method mainly extracts the spatiotemporal correlation features of long-term sequence data, combining the advantages of CNN and LSTM models. However, the single ConvLSTM network model has a major shortcoming [24]. On the one hand, the extraction of spatiotemporal features of pollutant and meteorological data lacks consideration of the impact of city locations on the prediction. On the other hand, as the number of ConvLSTM layers increases, the model will have more problems with network degradation and training costs will increase rapidly. Therefore, it is difficult for Att-ConvLSTM to overcome the above two problems.

We believe that artificially obtained original input data contain insufficient features for learning by a deep network. Further, during network training, the feature information is gradually lost with increased network layer depth. Therefore, we propose a new autoencoder network structure, the anti-autoencoder, and employ a feature expansion method based on stacked anti-autoencoders. The spatial relationship of the pollutant concentration and meteorological data is incorporated through use of a Gaussian function to calculate effective weights for each city. The final prediction is obtained using an LSTM network, as this technique can solve long-term dependence problems and can yield superior pollutant concentration predictions in time series.

3. Problem formulation

3.1. Time series problem

In this study, we construct the FDN-learning model combined with stacked anti-autoencoders, a Gaussian function layer, and an LSTM. The stacked anti-autoencoders are used for dimension extension and to obtain additional information. The Gaussian function layer is used to calculate the spatial correlation between the surrounding and target cities, and for weighted summation of the stacked anti-autoencoder output. LSTM is used to extract time series features, because the present pollutant concentration is affected by past pollutant concentrations and meteorological factors. The time series prediction model is illustrated in Fig. 1.

In Fig. 1, The inputs of FDN-learning are the multi-city pollutant concentration and meteorological data $x = (x_{t-r+1}, \ldots, x_i, \ldots, x_t)$, $x_i = (city_1^i, \ldots, city_j^i, \ldots, city_s^i)$, $city_j^i \in R^m$ with time series features (x_i, $city_j^i$, s, and m respectively represent the input of the i-th moment, input of city j of the i-th moment, number of the cities, and pollutants and meteorological factors), over the last r hour. The output is the pollutant concentration of the future n hour. Unlike the traditional pollutant prediction model, this study combines the advantages of stacked anti-autoencoders, a Gaussian function layer, and an LSTM to design a three-level architecture for FDN-Learning.

The base consists of stacked anti-autoencoders, each stacked anti-autoencoder network encodes the features to increase the dimension and mine the feature relationships. At the end of this layer, the stacked anti-autoencoders extracts high-level hidden features $out = (out^{t-r+1}, \ldots, out^i, \ldots, out^t), out^i \in R^{s*d}$ (d represents the hidden feature dimension). The second level is the Gaussian function layer, which uses Gaussian function to calculate the influence between cities, and the output of each stacked anti-autoencoder is weighted and summed to produce a one-dimensional spatial correlation feature vector vec_i at time i. LSTM

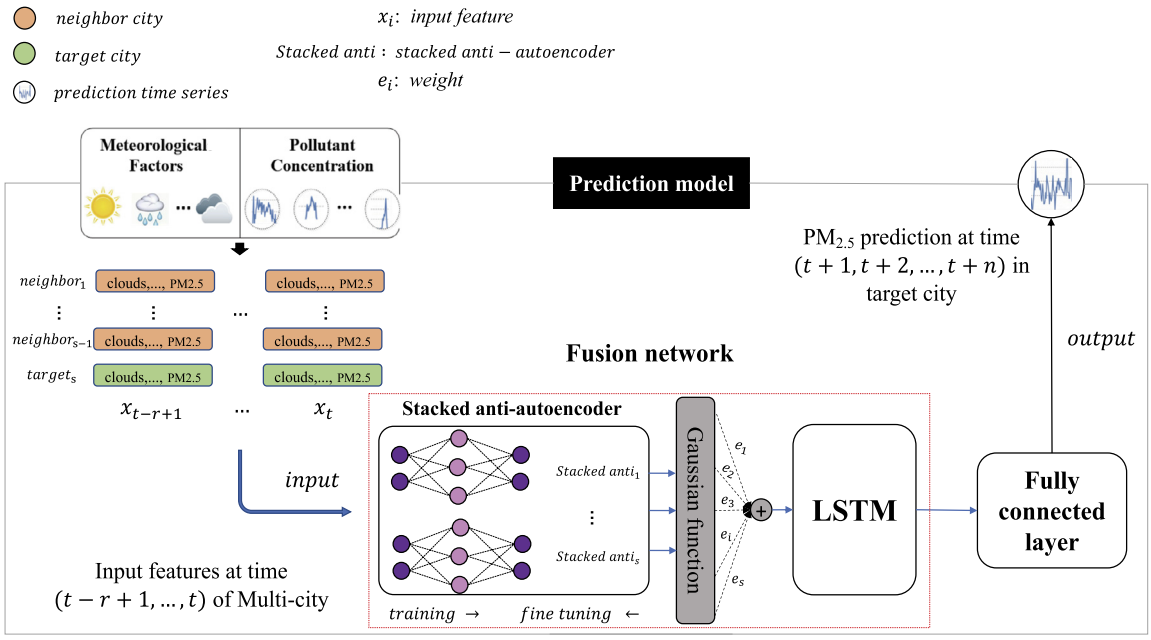


Fig. 1. FDN-Learning time series prediction model framework. The left side corresponds to the input time series data x, the center shows the predictive model transmission process, and the right side corresponds to the output time series prediction result $\hat{y}$. (For interpretation of the colors in the figures, the reader is referred to the web version of this article.)

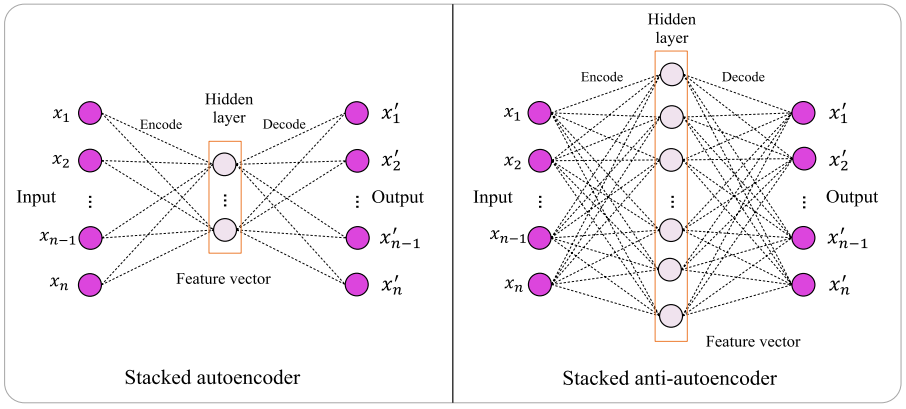


Fig. 2. (a) Three-layer stacked autoencoder. (b) Three-layer stacked anti-autoencoder. The feature vector is the target vector obtained by the encoding layer.

is set as the top of the fusion network and is designed to solve long-term dependence problems, being effective for extracting the input time series features $vec = (vec_{t-r+1}, \ldots, vec_i, \ldots, vec_t)$ and uses fully connected layer to produce the final prediction result $\hat{y} = (\hat{y}_{t+1}, \ldots, \hat{y}_{t+n})$.

3.2. Stacked anti-autoencoders

An autoencoder is a kind of multi-layer perceptron with a special form. It is trained by setting the output equal to the input, with the aim of reducing the reconstruction error between the input and output. Autoencoders can learn features in an unsupervised manner. Fig. 2(a) shows the simplest traditional autoencoder network structure, while Fig. 2(b) presents the anti-autoencoder network structure proposed in this paper.

The main difference between the two coding networks is the hidden layer structure of the network. The stacked autoencoder compresses and decompresses the input features, and its purpose is to compress the input feature vector through the encoder and restore the compressed feature vector to the maximum possible extent through the decoder. The advantage of the stacked autoencoder is that it can reduce the calculation cost and maximize the input feature value reduction. However, the stacked anti-autoencoder performs dimension expansion and decompression of the input features. The stacked anti-autoencoder expands the feature vector on the basis of the autoencoder. The advantages of the stacked anti-autoencoder are that it can (1) reduce the feature information loss in the transmission process, and (2) more fully mine and express the correlations between the input data. The similarities and differences between the two networks are discussed below.

The training processes of both networks can be divided into two steps: unsupervised pre-training and supervised fine-tuning. In the first step, each hidden layer is trained by a greedy algorithm. The first-layer output is expected to be as similar as possible to the input layer. Then the weights and bias of the link between the input and first layer can be obtained. The information of the first layer is the encoded denotation for the input. The original data should be recoverable through decoding, that is, the feature reconstruction process. Thus the reconstruction error is fed back to adjust the weights and bias. Next, the first-layer information is used to train the second layer in the same manner. The pre-

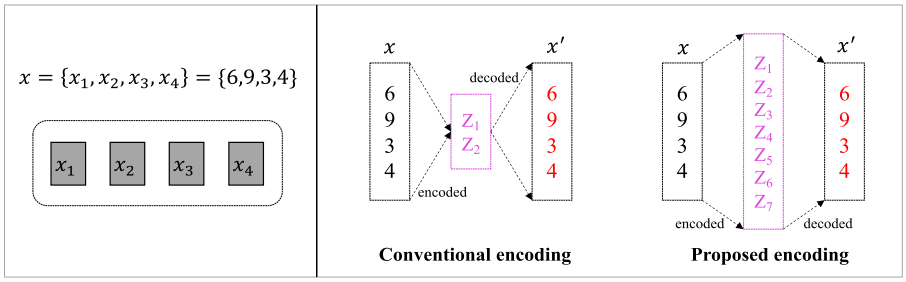


Fig. 3. Comparative example of conventional and proposed encoding methods.

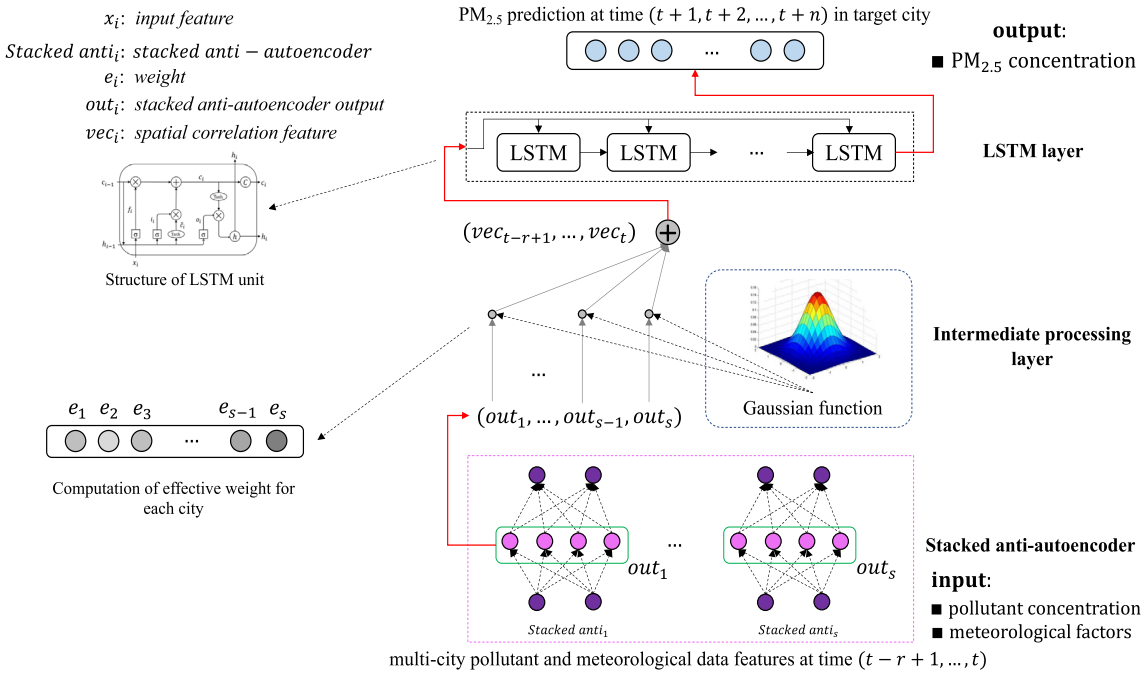


Fig. 4. FDN-Learning internal structure. In the figure, from bottom to top, it is composed of stacked anti-autoencoders, Gaussian function and LSTM. The red solid line represents the feature transfer direction, and the gray solid line represents the feature weighting process.

training procedures are repeated until the final output is obtained and the parameters of each layer are definite.

The second step is fine-tuning based on the reconstruction error between the input and output to obtain the global optimization. Here, we define x as the input vector, which will be encoded later, and x' as the decoded output vector. W_θ is the set of network weights, and E and D represent encoding and decoding functions, respectively. The pre-training process is implemented to reduce the reconstruction error $J(W_\theta)$,

$$J(W_\theta) = L(x, x') = L(x, D(E(x, W_\theta), W_\theta)) = \sqrt{\frac{\sum_i^N (x_i - D(E(x_i, W_\theta), W_\theta))^2}{N}} \quad (1)$$

where L represents pre-training loss function; i is the sample number, and N is the training set size.

As noted previously, the conventional stacked autoencoders discussed above are not used in this work. Conventional stacked autoencoders exploit feature compression. However, the existence of many external factors causes a lack of sufficient features in the original datasets. Then it becomes difficult for the deep network to learn the data features and adequately mine the feature relationships. To overcome this problem, our proposed approach encodes the original features but expands the feature dimensions by adding hidden layer nodes. Then the encoded information is decoded to determine whether the original features can be recovered. Fig. 3 presents an example comparing the two encoding methods, where x is the input data. The conventional approach targets compression to reduce data redundancy. The proposed method considers the shortcomings of the original dataset and can learn features adequately, while also deeply mining the relationship between the features.

4. Proposed method

In the above Section 3, we have introduced the FDN-Leaning prediction model framework and related problem descriptions. In this Section, we will describe the training process of each part of the FDN-Leaning prediction method in detail. Fig. 4 shows the structure of the FDN-learning model. The training process can be divided into three steps,

(1) The bottom layer is a parallel extraction layer for the multi-city pollutant and meteorological data features, and is composed of s stacked anti-autoencoders. The input of each stacked anti-autoencoder at time i is the pollutant and meteorological data in a city, and the input form is $city_j^i \in R^m$, as shown in Fig. 1 and Fig. 4, represents the pollutant and meteorological data of city j (e.g., target city). This layer

outputs s one-dimensional expansion feature vectors $out^i = (out^i_1, \ldots, out^i_j, \ldots, out^i_s), out^i_j \in R^d$ at time i.

(2) The second layer is an intermediate processing layer consisting of a Gaussian function. This function calculates the Gaussian function values of the geographic locations of the surrounding-city monitoring stations and target-city monitoring station, and the output out^i_j of each stacked anti-autoencoder is weighted and summed to produce a one-dimensional spatial correlation feature vector $vec_i \in R^d$ at time i.

(3) The top layer is the LSTM layer, which performs timing feature extraction on the Gaussian function layer output $vec = (vec_{t-r+1}, \ldots, vec_i, \ldots, vec_t)$ and inputs the final hidden state to the fully connected layer to produce the final prediction result, $\hat{y} = (\hat{y}_{t+1}, \ldots, \hat{y}_{t+n})$.

4.1. Stacked anti-autoencoder training

The first step is to obtain trained stacked anti-autoencoders. The method is almost identical to that described in Section 3.2. In the experiments performed in this study, past meteorological and pollution data was taken as input, in the form of one-dimensional vectors corresponding to s cities (one target city and $s-1$ neighboring cities). These vectors were individually input to the model for pre-training of the stacked anti-autoencoders. Therefore, the stacked anti-autoencoders could adequately learn the data features of each city. Although the many external factors caused the problem of insufficient features in the original dataset, the stacked anti-autoencoders could encode the features to increase the dimension and mine the feature relationships. Finally, s expanded output vectors were obtained. These new vectors represented the original input well, as it was almost restored after the decoding validation.

In the proposed method, the hidden layers of the base network are set so as to increase the input dimension and obtain the hidden relationships between the data features. Each hidden layer of the stacked anti-autoencoder is individually trained by a greedy algorithm. The output of the first hidden layer is expected to be as similar as possible to the input layer. Then a BP algorithm is used to fine-tune this layer. The weights and bias of the link between the input and first hidden layer are discussed below. The trained first hidden layer is next input to the second hidden layer, and the second layer is trained in the same way. The input layer is removed from the training, as it is meaningless at that stage. The pre-training procedures are repeated until the final output is obtained and the parameters of each layer are determined. The weights are updated during the training using the following formula:

$$W^k_\theta = W^k_\theta - \alpha \frac{\partial}{\partial W^k_\theta} J(W^k_\theta), k = 1, 2, 3, ..., l \tag{2}$$

Where α is the learning rate, k represents the layer of stacked anti-autoencoders, and l represents the number of layers.

After the stacked anti-autoencoders are trained, their outputs are transferred to the intermediate processing step.

4.2. Intermediate processing of stacked anti-autoencoder output

The final outputs of the stacked anti-autoencoders are s high-dimensional vectors. The s input vectors are individually trained so that the network can adequately learn the features of each city. These vectors can precisely represent the low-dimension inputs. In the experiments performed in this study, $s-1$ neighbor cities close to the target city were selected, because the pollutant concentration is influenced by the spatial relationship. The effective degrees of the neighbor cities relative to the target city were determined based on distance (Fig. 5). The outputs of the stacked anti-autoencoders for the s cities were merged into one-dimensional vectors in time series based on distance using the Gaussian function, before being input to LSTM. Therefore, the spatial effect on the pollutant concentration was enhanced and the complexity generated by processing s matrices was avoided.

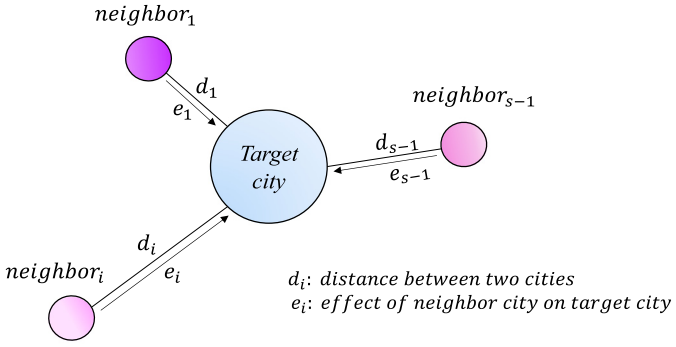


Fig. 5. Spatial effect on target city.

r one-dimensional vectors were output during the time period $(t-r+1, \ldots, i, \ldots, t)$ for each city. The Gaussian function was used to calculate the effective weights for each city based on its distance to the target city. The Gaussian function has the peculiarity that the weight is 1 when the distance between two points is 0, and tends to 0 as the distance increases. This process is illustrated in Fig. 6.

In the proposed method, the Gaussian function, which processes the relationship between the distance and determines the weight, is used to calculate the effective weights (Fig. 6) as

$$e(x, y) = e(x) * e(y) = A^2 * \exp(-B) \tag{3}$$

$$A = \frac{1}{\sigma\sqrt{2\pi}}, B = \frac{(x-\mu)^2}{2\sigma^2} + \frac{(y-\mu)^2}{2\sigma^2} \tag{4}$$

where (x, y) is the geographical coordinates (latitude and longitude) of the non-target monitoring site, and (x_0, y_0) is the geographical coordinates of the target monitoring site. Ultimately, the e values are obtained, and the latter are labeled $(e_1, e_2, \ldots, e_s)$. Then, the new input vec_i can be expressed as

$$vec_i = e_1 * out^i_1 + e_2 * out^i_2 +, \ldots, +e_s * out^i_s \tag{5}$$

where $(out^i_1, out^i_2, \ldots, out^i_s)$ are respectively the outputs of the stacked anti-autoencoders at time i for the neighbor cities and the target city.

4.3. LSTM training process

The new LSTM input containing the spatial features is obtained using the method described in the previous section. The temporal features are extracted by LSTM, which is advantageous for processing long-term time dependence problems and outputs the final prediction result. Assuming that c_i is the cell state, h_i is the cell output, and vec_i is the spatial feature of the input pollutant and meteorological data at time i. Additionally, $*$ denotes the Matrix multiplication, $\circ$ denotes the Hadamard product, and i_i, f_i and o_i respectively represent the input gate, forge gate and output gate at time i. The training process at each time series of LSTM can be expressed by the following formulas:

1) LSTM selectively forgets the feature information of cell state at time i,

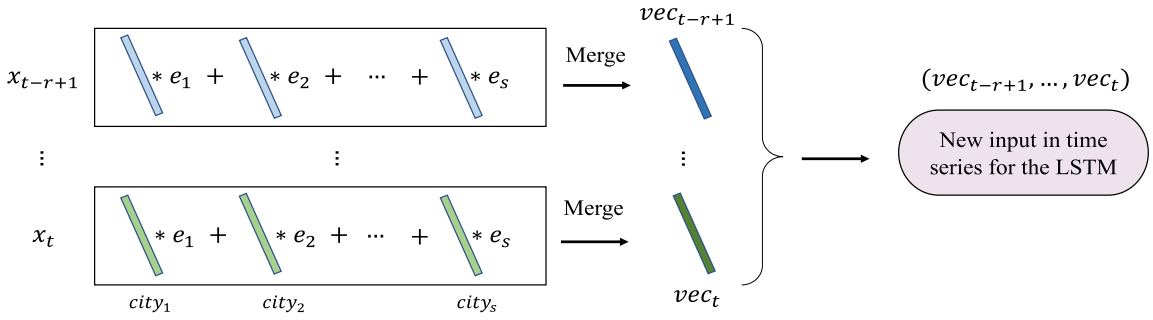


Fig. 6. Vector merging process.

$$\begin{cases} f_i = \sigma(W_f * vec_i + W_f * h_{i-1} + W_f * c_{i-1} + b_f) \\ c'_i = f_i \circ c_{i-1} \end{cases} \tag{6}$$

We need to selectively forget the information in the memory unit c_{i-1}. Therefore, we choose the σ (*sigmoid*) function as the activation function of the forget gate. LSTM forgets the feature information with a small contribution in the memory unit c_{i-1} according to the current input $[h_{i-1}, vec_i, c_{i-1}]$. The σ function assigns different weight values to the feature elements in each dimension of the memory unit c_{i-1}, and the value is constrained between [0.0, 1.0], as shown in formula (6) (if the weight value is 1.0, the corresponding dimension feature information is all saved. On the contrary, if the weight value is 0.0, the corresponding memory information is completely forgotten). That is, we use the *sigmoid* function to weight the input features $[h_{i-1}, vec_i, c_{i-1}]$, a small weight indicates that the importance of the feature information is less, and a large weight indicates that the feature is more important. The weighting process is completed through training of the neural network, and the weight ranges are between [0.0, 1.0]. For example, $f_i \circ c_{i-1} = [0.0, 0.4, 0.6, 1.0] * [1, 2, 3, 4]$, where feature element 4 is important information.

By multiplying the output of the memory unit c_{i-1} and the forget gate f_i, a part of the memory information is forgotten. After part of the state information of the memory unit is forgotten, LSTM needs to update the information of the memory unit according to the current input gate i_i.

2) LSTM selects important information from the input features that is used to update the state cell,

$$\begin{cases} \tilde{c}_i = tanh(W_{\tilde{c}} * vec_i + W_{\tilde{c}} * h_{i-1} + W_{\tilde{c}} * c_{i-1} + b_{\tilde{c}}) \\ i_i = \sigma(W_i * vec_i + W_i * x_{i-1} + W_i * c_{i-1} + b_i) \\ c_i = c'_i + i_i \circ \tilde{c}_i \end{cases} \tag{7}$$

In the above equations, $\widetilde{c}_i$ represents the initial feature used to update the information of the memory unit c'_i. The initial feature is calculated by the *tanh* and the input $[h_{i-1}, vec_i, c_{i-1}]$ of the LSTM, as shown in formula (7). The implementation principle of the input gate i_i is the same as the forget gate f_i, as shown in formula (7). The function of input gate i_i is mainly to assign different weight values to the elements in each dimension of $\widetilde{c}_i$, and to select the important feature information for updating the memory unit c'_i (if the weight value is 1.0, the corresponding dimension feature information is all saved).

3) Finally, it determines what LSTM must output,

$$\begin{cases} o_i = \sigma(W_o * vec_i + W_o * h_{i-1} + W_o * c_{i-1} + b_o) \\ h_i = o_i \circ tanh(c_i) \end{cases} \tag{8}$$

The output gate o_i is based on the input $[h_{t-1}, vec_i, c_{i-1}]$ at the current time, and selects some important feature information from the memory unit c_i for output at the current time. The σ function of output gate is mainly to assign different weight values to the elements in each dimension of c_i and to select the important feature information for output. The output of the LSTM is h_i and we enter the results of the calculation into the fully connected layers to generate the predicted value.

After LSTM has been trained and the final result has been obtained, the entire FDN-learning model is fine-tuned via stochastic gradient descent to attain global optimization. An elastic net, which is a regularization method explained in our previous study [36], is used to avoid the over-fitting problem and to enhance the generalization ability of the prediction model.

5. Experimental results

5.1. Data description

The experiment used historical pollutant concentration and meteorological data from monitoring stations in 14 cities collected from May 13, 2014, to May 30, 2018 (codes and data sample: https://github.com/zouguojian/FDN-Learning). In this paper, the selection of city sites, 14 cities (Shanghai, Nanjing, Suzhou, Nantong, Wuxi, Changzhou, Zhenjiang, Hangzhou, Ningbo, Shaoxing, Huzhou, Jiaxing, Taizhou, and Zhoushan) with rapid economic development in the Yangtze River Delta region centered on Shanghai (Fig. 7). The geographical location of these cities is closest to Shanghai, and the spread of pollutants is more likely to affect each other. We selected 16 pollutants and meteorological factors: AQI, $PM_{2.5}$, PM_{10}, SO_2, NO_2, O_3, CO, Temp (temperature), Hum (humidity), air pressure, wind direction, wind speed, clouds, maximum temperature, minimum temperature and Conds (meteorological conditions).

In our experiment, we selected 70% of the data as the training set, 15% as validation set, and the remaining 15% was used as the test set. The specific method of dividing the data in this study is as follows: first, we divide the data set uniformly according to a given window length L and a moving step size of S, and finally the total number of samples obtained is $Total = ((D - D * 0.15) - L)/S$; then, we scramble the Total samples, select 82% of them as the training set and 18% as the validation set. In addition, 15% of D is used as the test set, which means that we extract 15% of the data from the original data set as the test set without disturbing it; finally, we define our division method as a generalized random method. Among them, the window length L represents the sum of the time sequence length of the input model and the target prediction sequence length, and D is the size of the original data set. The missing values of the air pollutant concentration and meteorological data set are filled by spatiotemporal interpolation [50]. This paper attempted to predict the future n hour pollutant concentration in the target city by using the pollutant and meteorological data in the past r hour.

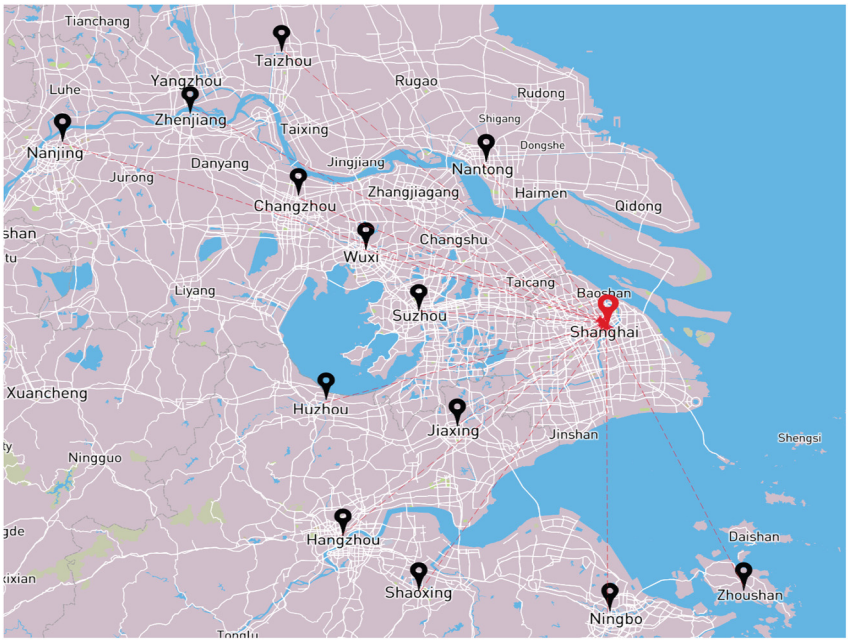


Fig. 7. Comprehensive utilization of air-pollutant data in multi-city monitoring stations. The black circles indicate the neighbor cities, the red circle indicates the target city, and the arrow indicates the possible impact of the surrounding city pollutants on the target city.

5.2. Experimental setup

5.2.1. Related definitions of FDN-learning

In the FDN-learning model, the loss function is used to measure the degree of inconsistency between the predicted value $\hat{y} = (\hat{y}_{t+1}, \ldots, \hat{y}_i, \ldots, \hat{y}_{t+n})$ and the real value $y = (y_{t+1}, \ldots, y_i, \ldots, y_{t+n})$. The loss function is given in (9):

$$loss = \sqrt{\frac{\sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2}{n}} + \frac{\lambda}{2}||W||_2^2 \tag{9}$$

where n is the length of the predicted sequence, y_i denotes the observed value of the pollutant concentration, $\hat{y}_i$ is the predicted value of the air pollutant concentration, λ is the regularization parameter, and W is the weight parameter of the FDN-learning. The loss function distributes the calculated error to all layers of the network through backpropagation and uses the stochastic gradient descent algorithm to adjust the weights in the network until the network converges.

The FDN-learning model presented in this paper was compared with other prediction models on the same dataset. The root mean square error (RMSE), mean absolute error (MAE), run time, and correlation coefficient (Corr) were used as metrics to evaluate the correctness and effectiveness of the proposed method. The experimental metrics were calculated using the following formulas:

$$\begin{cases} RMSE(y, \hat{y}) = \sqrt{\frac{\sum_{i=1}^{T}(y_i - \hat{y}_i)^2}{T}} \\ MAE(y, \hat{y}) = \frac{1}{T}\sum_{i=1}^{T}\left|y_i - \hat{y}_i\right| \\ Corr = \frac{Cov(y, \hat{y})}{\sqrt{var[y] * var[\hat{y}]}} \end{cases} \tag{10}$$

where y_i is the observed value, $\hat{y}_i$ denotes the predicted value, T is the test set size, $Cov(y, \hat{y})$ is the covariance of y and $\hat{y}$, and $var[y]$ and $var[\hat{y}]$ represent the variance of y and $\hat{y}$, respectively.

5.2.2. Training

The hyperparameters in our FDN-learning are determined during the training process, that is, the best performance model is selected on the validation set through the RMSE. We manually specify the hyperparameter ranges: learning rate {0.01, 0.005, 0.003, 0.001, 0.0005}, dropout rate {0.0, 0.1, 0.2, 0.3, 0.4, 0.5}, regularization parameter {0.1, 0.01, 0.001, 0.0001} and decay rate {0.99, 0.95, 0.90, 0.85}. For different datasets, we have found that the following setting work well: set the dropout to 0.5, decay rate to 0.99, regularization parameter to 0.0001 and learning rate to 0.0005 for FDN-learning. When using the comparison models, these settings still work well. We implemented FDN-learning in Tensorflow, and train all models using the SGD optimizer with batch size 128.

5.2.3. Evaluation

The setting of hyperparameters in this study is based on the results of many experiments, leading to the final selection of the optimal set of hyperparameters. The validation set used in this study is closely related to the training stage, and after each epoch, the RMSE and MAE of the prediction model on the validation set are calculated. Therefore, the optimal model is selected based on the model error calculated on the validation set. The specific process is as follows: for each experiment, the number of epochs selected was 100. After training an epoch, we tested the trained model on the validation set. If the RMSE and MAE of the prediction model on the validation set became smaller, we updated and saved the model parameters. After many parameter adjustments and experiments, when the prediction effect of the prediction model on the validation set was optimal, the training ended. Finally, get the prediction result by iterating all the samples in the test set.

5.3. Parameter setting

To determine the optimal structure of FDN-learning, we vary the number of nodes in each layer of the network and compare the prediction results of FDN-learning with different structures through experiments. The prediction performances of the FDN-learning with different number of nodes in each layer are shown in Table 1. In the experiments performed in this study, the values of r and n were 3 and 1, respectively.

According to the experimental results in Table 1, it can be concluded that the optimal prediction performance is obtained when

Table 1
Predictive performance of FDN-learning with different nodes [3-1 h].

Nodes	RMSE	MAE	Corr	Run time
(16-32-64)-(32)	4.623	3.212	0.984	**5 min 6 s**
(32-64-128)-(32)	4.900	3.364	0.983	7 min 9 s
(64-128-256)-(32)	5.269	3.688	0.980	11 min 23 s
(128-256-512)-(32)	5.491	3,768	0.978	25 min 13 s
(16-32-64)-(64)	4.615	3.157	0.984	5 min 12 s
(16-32-64)-(128)	4.401	3.079	0.986	7 min 35 s
(16-32-64)-(256)	**4.326**	3.307	**0.987**	11 min 23 s
(16-32-64)-(512)	4.351	**2.997**	0.986	25 min 16 s

Table 2
Model parameters.

Layer name	Parameter	Values
Anti-autoencoders	[layer nodes] × number of autoencoders	[16,32,64]×s
Gaussian layer	[layer nodes] × number of layers	–
LSTM	[layer nodes] × number of layers	[256]×1
Full connected layer	[layer nodes] × number of layers	[256]×1 [128]×1 [1]×1
–	Batch-size	128
–	Learning rate	0.0005
–	Decay rate	0.99
–	Dropout	0.5
–	Epoch	100
–	μ	0
–	σ	1
–	λ	0.0001
–	Training method	SGD

Table 3
Performance comparison of different methods [3-1 h].

Model	RMSE	MAE	Corr
CAMx [60]	34.454	–	0.712
CMAQ [6]	34.087	–	0.708
NAOPMS [45]	36.649	–	0.690
WRF-Chen [38]	37.316	–	0.683
SVM [39]	25.820	18.415	0.858
SVR [51]	23.564	16.001	0.869
MLR [11]	19.023	13.284	0.934
HMM [40]	22.261	15.034	0.882
XGBoost [55]	21.090	14.964	0.887
BP [7]	18.915	13.043	0.934
RNN [5]	15.932	11.715	0.962
LSTM [34]	15.721	9.895	0.966
Bi-LSTM [56]	13.245	9.654	0.969
EDSModel [57]	12.628	9.172	0.972
ConvLSTM [24]	13.041	9.234	0.970
CNN-LSTM [36]	11.366	8.221	0.974
GC-LSTM [35]	10.478	7.503	0.975
Att-ConvLSTM [47]	11.476	8.321	0.974
FDN-Learning	**4.326**	**3.307**	**0.987**

Table 4
Air pollutant concentration prediction over different time periods [48-10, 20, 30, 40 h].

Model	0-10 h	0-20 h	0-30 h	0-40 h
Att-ConvLSTM [47]	22.053	25.010	26.146	26.768
CNN-LSTM [36]	22.004	24.987	26.087	26.580
GC-LSTM [35]	20.865	23.346	25.002	25.764
FDN-Learning	**16.785**	**21.993**	**23.295**	**24.016**

the number of layers is (16–32-64)–(256), combining the time cost and the accuracy of the model.

In the experiment, dropout was used as a general trick to avoid model overfitting. According to historical experience and research results in the field of deep learning, the effect is obvious when the value of the training stage is 0.5. Therefore, in the training stage of different prediction models, the value of dropout is 0.5. In the verification and testing stage, for each model, the value of dropout is 1.0. After the experiments, the parameters used for FDN-learning model testing are shown in Table 2.

5.4. Experimental comparison

5.4.1. Prediction for the next hour

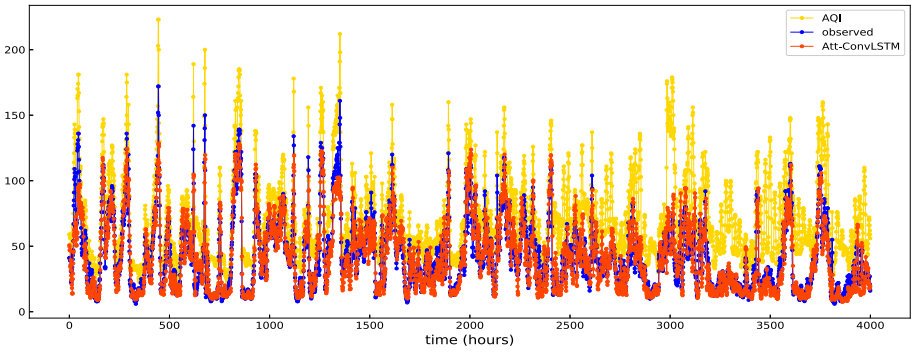
To show the advantages of the FDN-learning model, Table 3 lists the RMSE values, MAE values, and Corr values for each model on the whole test set.

Fig. 8 shows the generalization ability of different models on the same test set. The length of Fig. 8's x-axis is 4000 hours, which means that 4000 consecutive hours were randomly selected in the test set to test the performance of the prediction model in this time period. This verification method is based on the method given in previous research papers [36,33,57], and the main purpose is to visualize the prediction effect of the model and highlight the prediction performance and fitting ability of the model. We combine the prediction of pollutant with the change of AQI, and describe the location of mutation points more scientifically through AQI. According to the description of Yi et al. [53], when the AQI value fluctuates sharply, the mutation point appears. Therefore, we combine the test results with the mutation points to further verify the superiority of our FDN-learning model. The blue curve represents the observed value, the red curve represents the predicted value and the yellow curve represents the AQI value. Owing to space considerations in this study, Fig. 8 only shows the experimental results of the four state-of-the-art prediction models, representing the fitting trends of the Att-ConvLSTM, CNN-LSTM, GC-LSTM, and FDN-learning models were tested on the whole test set.

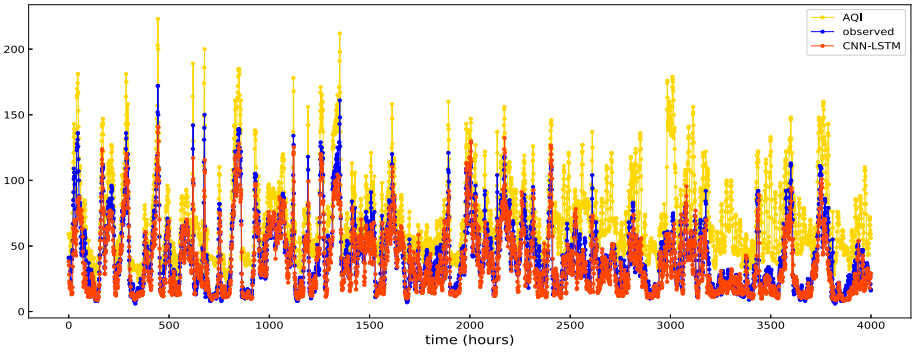
To demonstrate the predictive performance of the FDN-learning model we chose, we compared it with the latest research results. We selected four prediction models, including the proposed FDN-learning. Fig. 9 depicts the prediction performance of different prediction models on the test set. The x-axis represents the observed value of $PM_{2.5}$ and the y-axis represents the predicted value of $PM_{2.5}$. The black line indicates the $y = \hat{y}$ function, and the black dots indicate the degree of deviation between the observed and predicted values. In the dispersion comparison, when the concentration of $PM_{2.5}$ is greater than 100 μg/m^3, the dispersion of Att-ConvLSTM is the largest, and that of FDN-learning model is the smallest, meaning that the prediction performance is the best. When the values of $PM_{2.5}$ are between 0 μg/m^3 and 100 μg/m^3, the dispersion degree of FDN-learning model is still the smallest. Fig. 9 shows that the FDN-learning predicted values are generally consistent with the observed values. In the correlation comparison, in the whole test set, the correlation coefficients of Att-ConvLSTM, CNN-LSTM, GC-LSTM, and FDN-learning are 0.974, 0.974, 0.975, and 0.987, respectively, which means that the correlation between predicted values and observed values of FDN-learning is the largest.

5.4.2. Prediction for the time-series

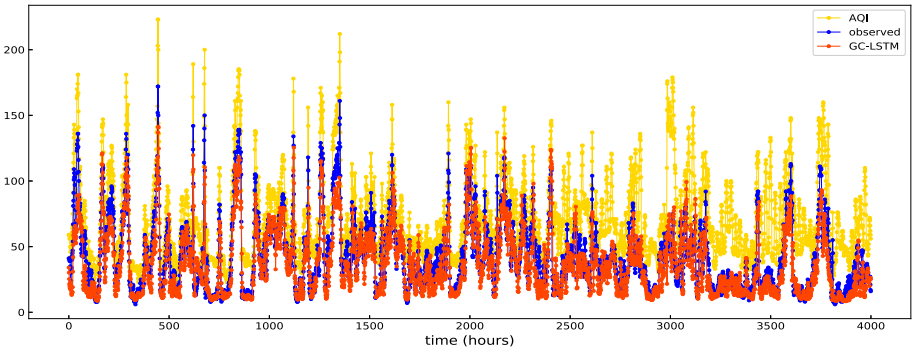
To further confirm the validity of the proposed prediction model, we used the past 48 hours of multi-city pollutant and meteorological data as input to predict the pollutant $PM_{2.5}$ concentrations over the different time periods. We compared the FDN-learning model with the state-of-the-art $PM_{2.5}$ prediction models: Att-ConvLSTM, CNN-LSTM, and GC-LSTM. Table 4 shows the RMSE of $PM_{2.5}$ concentration prediction values over the different time periods.



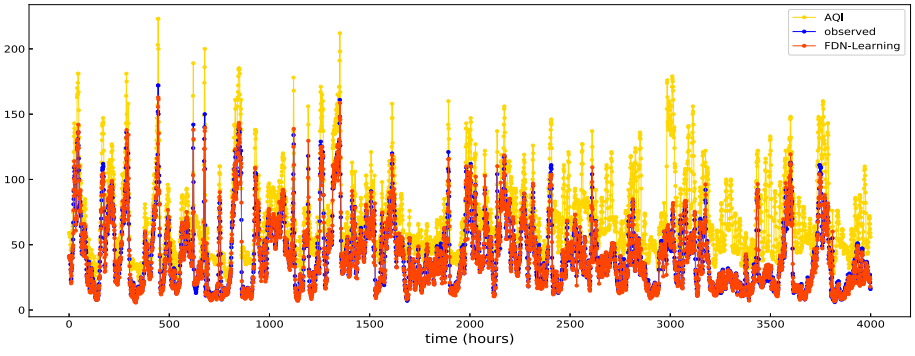
(a)



(b)



(c)



(d)

Fig. 8. Fitting trends of the different models. (a)-(d) Att-ConvLSTM, CNN-LSTM, GC-LSTM, and FDN-Learning.

To further demonstrate the fitting performance of FDN-Learning on the test set, we predicted the pollutant concentration in the future $n = 40$ hours. Fig. 10 shows the predicted and observed changes in $PM_{2.5}$ over the different time periods (with randomly selected samples from different time periods on the test set).

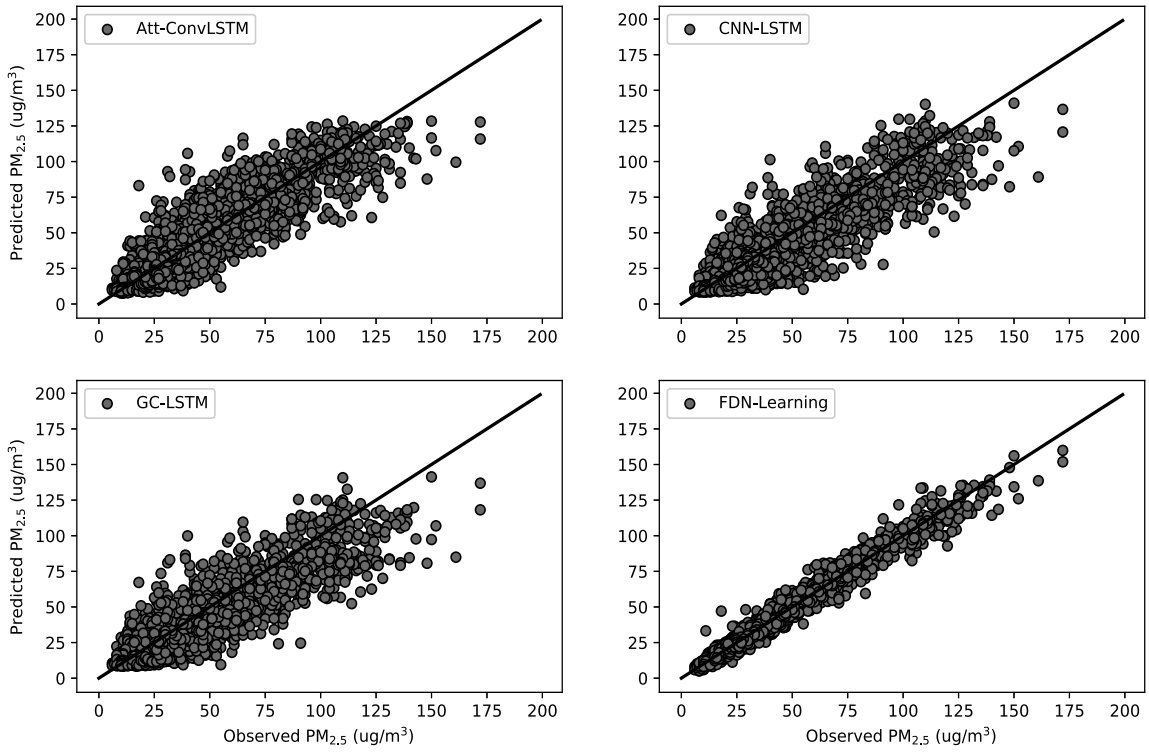


Fig. 9. Degree of fit between the observed and predicted values on the test set.

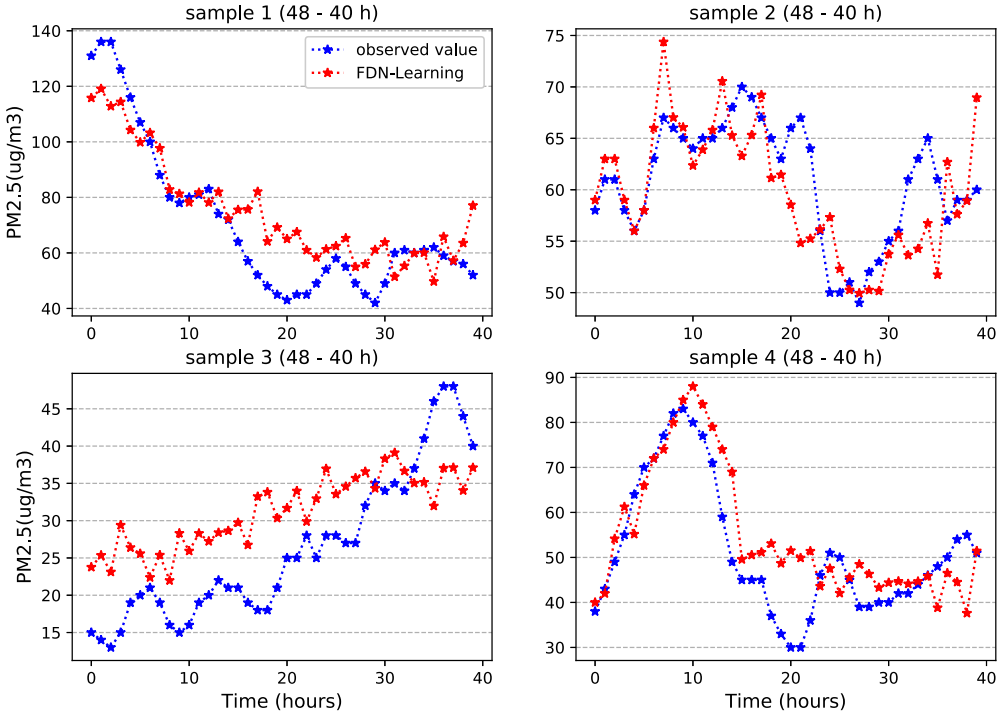


Fig. 10. Prediction of target city pollutant concentration over the different time periods. The blue curve represents the observed values, the red curve represents the predicted values.

5.5. Spatial weighted prediction performance

In addition, to prove the influence of the surrounding cities on the prediction accuracy for the air pollutant concentration in the target city, the model prediction efficacy for two input cases was compared: one considering the target city data only, and the other incorporating the additional data from the surrounding cities. FDN-learningS: the input to the model is only the data for a single target city and the bottom layer of the model is a single stacked anti-autoencoder. FDN-learning: the model input is the data for all cities and the bottom layer of the model is s parallel stacked anti-autoencoders. FDN-LearningW: represents a special form of the FDN-Learning method, without Gaussian function layer. The experimental results are presented in Table 5.

Table 5
Effect of additional pollution data [3-1 h].

Model	RMSE	MAE	Corr
FDN-LearningS	7.047	4.883	0.980
FDN-LearningW	5.152	3.444	0.981
FDN-Learning	**4.326**	**3.307**	**0.987**

Table 6
FDN-Learning generalization ability [3-1 h].

City	RMSE	MAE	Corr
Shanghai	4.326	3.307	0.987
Changzhou	4.102	2.744	0.974
Wuxi	5.742	3.641	0.973
Suzhou	5.142	3.640	0.982
Jiaxing	5.962	3.827	0.972

5.6. Predictive model generalization ability

In order to verify the generalization ability and effectiveness of the FDN-learning model proposed in this paper, we applied the proposed FDN-learning model to other cities in China for pollutant concentration prediction. We use monitoring data from multiple cities to train the prediction model proposed in this paper, and test the generalization ability of the model in five cities in China. The experimental results are presented in Table 6.

6. Discussion

6.1. Comparison with previous prediction models

6.1.1. Analysis of the prediction results for the next hour

Table 3 shows that, compared with the six single models and nine traditional methods, the CNN-LSTM, GC-LSTM, Att-ConvLSTM, and FDN-Learning have better predictive results because all four can better handle long-term sequence dependency problems with spatial features. Comparing the prediction results in Table 3 of CNN-LSTM and GC-LSTM, the prediction accuracy of FDN-Learning is higher than that of GC-LSTM and CNN-LSTM, which proves that the combined network of stacked anti-autoencoders and Gaussian function has better spatial feature ability to extract pollutant and meteorological data than deep CNN and graph convolutional (GC) neural network. Its RMSE, MAE, and Corr attain the optimal values of 4.326, 3.307, and 0.987, respectively. Next, Comparing the prediction results in Table 3 of Att-ConvLSTM and FDN-Learning, the prediction accuracy of FDN-Learning is higher than that of Att-ConvLSTM, which proves that deep FDN-Learning has better spatiotemporal feature ability to extract pollutant and meteorological data than spatiotemporal attention method Att-ConvLSTM. However, using only the LSTM model to extract the temporal and spatial features of the complex pollutant and meteorological data, it is difficult to deeply extract the spatiotemporal features of time series. Therefore, this study combines the advantages of stacked anti-autoencoders, Gaussian function and LSTM, and proposes a new type of prediction framework: the FDN-Learning model. The experimental results in Table 3 also confirm that FDN-Learning is very effective for the prediction of $PM_{2.5}$.

In this paper, 4000 consecutive test samples were randomly selected and presented in the experiment in the form of graph, as shown in Fig. 8 and Fig. 9. When the $PM_{2.5}$ pollution source concentration is unstable, particularly when the concentration value is greater than 100 $\mu g/m^3$, the prediction results of the comparison models could not follow the actual trend and showed a rather disordered pattern. This also reflects the fact that, in terms of the current $PM_{2.5}$ concentration prediction task, it is still difficult for the model to make accurate predictions. Furthermore, the predictions and observations of the proposed FDN-Learning model are almost coincident and have a good fitting effect on the mutation of $PM_{2.5}$ concentration, such as the 165th hour, 444th hour, 1350th hour, etc., as shown in Fig. 8.

Combining the fitting ability of each model in Fig. 8 and Fig. 9, we reach the following conclusions: (1) For the Fig. 8, we can get that the prediction performance of the FDN-Learning is better than the comparison models, and it is suitable for prediction tasks with sudden changes in pollutant concentration; (2) For the Fig. 9, we can get that compared with the comparison models, FDN-Learning can accurately predict high concentrations of $PM_{2.5}$, so that the predicted value and the observed value are highly consistent; (3) Combining the experimental results in Fig. 8 and Fig. 9, we can intuitively see that for mutation points, the $PM_{2.5}$ concentration is generally relatively high, and the number of mutation points is relatively small. This mainly reflects that in the general data set, the number of samples at mutation points is small, which leads to the problem of uneven data distribution. This phenomenon has caused the problem of insufficient learning of the predictive model, that is, it is difficult to learn the changing regularity of pollutant concentration under sudden changes. Therefore, this is also the reason why some models are difficult to fit in the case of sudden pollutant concentration.

Based on the above experimental results, our analysis result is that the FDN-Learning proposed in this paper tightly grasps the spatiotemporal characteristics of pollutants. In terms of data, we consider the impact of pollutants and meteorological factors in multiple cities on the target city in the pollutant concentration prediction task; In terms of the model, we utilize the stacked anti-autoencoders, Gaussian function and LSTM as the spatiotemporal feature extractor, and make full use of the advantages of the networks in feature extraction. Therefore, the characteristics of our prediction model are as follows: on the one hand, in large samples D_1 with small vibration amplitude of pollutant concentration, the changing regularity of pollutant concentration in historical data can be fully learned; on the other hand, in small samples D_2 with large fluctuations of pollutant concentration, we utilize the advantages of the FDN-Learning to learn the changing regularity of pollutant concentration, which can solve the problem that it is difficult to accurately predict the mutation of pollutants in the target city (training set $= D_1 + D_2$). The ability of the FDN-Learning to predict $PM_{2.5}$ concentration is verified in this experiment.

6.1.2. Analysis of the prediction results for the time-series

As shown in Table 4, when we compare the average prediction errors of the CNN-LSTM, GC-LSTM, Att-ConvLSTM, and FDN-Learning models for different prediction time periods, the prediction errors of CNN-LSTM, GC-LSTM, and Att-ConvLSTM are larger than the FDN-Learning model, meaning that FDN-Learning has the highest prediction accuracy. Fig. 10 shows the performance of the FDN-Learning model in predicting pollutant time-series for the 40-hour time period, that is, four test samples were randomly selected from the test set as the reference basis for the analysis of the experimental results. From Fig. 10 we can see that the trends indicated by the blue observation curve and the red prediction curve are consistent. The experiments verified that for the long-term prediction of pollutant concentration, the time-series predicted of FDN-Learning model has wide application value. Therefore, we can improve the accuracy of pollutant prediction by considering combining the time-series of pollutant concentration predicted by the FDN-Learning model with the state-of-the-art prediction methods.

6.2. Analysis spatial weighted prediction performance

Combining the experimental results in Table 3 and Table 5, some useful findings can be summarized. First, compared with existing pollutant prediction models, excluding FDN-learningW and

FDN-learning, the FDN-learningS model proposed in this paper yielded superior prediction results. This indicates that the stacked anti-autoencoder can better extract the hidden distribution features of the pollutant concentration and meteorological data, with RMSE, MAE, and Corr values that can reach 7.047, 4.883, and 0.980, respectively.

Second, compared with the FDN-learningW model, FDN-learning performs better. This proves that the pollutant concentration of the target city is affected by the location of surrounding cities. Considering the influence of spatial location on the pollutant concentration in the target city can improve the prediction performance of the model.

Third, FDN-learning achieved superior air pollutant concentration prediction compared to FDN-learningS and FDN-learningW. Thus, according to the experimental results shown in Table 5, fully considering the hidden distribution features of pollutant and meteorological data, as well as the impact of location-weighted surrounding cities on the pollutant concentration of the target city, can effectively improve the prediction performance of the model. Therefore, the FDN-learning model proposed on the basis of combining stacked anti-autoencoders, Gaussian function, and LSTM can effectively improve the accuracy of pollutant concentration prediction in the target city.

6.3. Analysis model generalization ability

Due to the differences in the environment of each city, the generalization of model on different cities is difficult. However, from the experimental results in Table 6, we can observe that for cities (Changzhou, Wuxi, Suzhou, and Jiaxing), FDN-Learning also has strong generalization accuracy. The FDN-Learning model based on stacked anti-autoencoders, Gaussian function, and LSTM not only solves the spatiotemporal problem in the pollutant concentration prediction task, also considers the impact of city locations on the pollutant concentration. Therefore, the performance of our proposed FDN-Learning is quite stable on the task of predicting the pollutant concentration in multiple cities. In the future work of pollutant concentration prediction, we can generalize the model proposed in this paper to more cities and improve the accuracy of pollutant concentration prediction task.

7. Conclusions

This study proposed the FDN-learning model for air pollutant concentration prediction that exploits the massive amounts of available pollutant concentration and meteorological data. FDN-learning model is composed of stacked anti-autoencoders, Gaussian function, and LSTM. The advantages of the proposed method can be summarized as follows,

(1) The feature expansion method of the anti-autoencoder can extract the hidden distribution features of the pollutant concentration and meteorological data obtained in a city;
(2) Taking the location information of the surrounding and target cities as influence factors for spatial correlation feature extraction of the pollutant concentration and meteorological data, the weighted summation of the stacked anti-autoencoder output results performed in the Gaussian function layer can further improve the model prediction performance;
(3) The LSTM addresses the time series problem, as the pollutants have time dependency. The trained stacked encoders and the untrained LSTM are fine-tuned to obtain the final model.

The experiments performed in this study proved that, compared to traditional models, the proposed FDN-learning model yields higher-accuracy predictions by fully extracting data correlations, and overcomes problems such as long-term dependency and spatial correlation. The time dependence of the pollutants is considered in this approach. Therefore, the proposed FDN-learning model overcomes the weaknesses of traditional machine learning methods and single classic networks and is valuable for practical application. Compared with the traditional machine learning methods and deep learning networks, the FDN-learning has been applied as one of the practical auxiliary models in the national urban air pollution monitoring and prediction tasks for many times, which shows good application effect and value. However, this paper does not fully consider how to generalize the model to cities with serious pollution, which will be solved in future research.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work is funded by National Natural Science Foundation of China (61572326, 61802258, 61702333), Natural Science Foundation of Shanghai (18ZR1428300), the Shanghai Committee of Science and Technology (17070502800).

References

[1] H. Akimoto, Global air quality and pollution, Science 302 (2003) 1716–1719.
[2] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, S. Savarese, Social lstm: human trajectory prediction in crowded spaces, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 961–971.
[3] L. Bonnet, The method of Gaussian weighted trajectories. III. An adiabaticity correction proposal, J. Chem. Phys. 128 (2008) 044109.
[4] E.K. Cairncross, J. John, M. Zunckel, A novel air pollution index based on the relative risk of daily mortality associated with short-term exposure to common air pollutants, Atmos. Environ. 41 (2007) 8442–8454.
[5] H. Chang-Hoi, I. Park, H.R. Oh, H.J. Gim, S.K. Hur, J. Kim, D.R. Choi, Development of a pm2. 5 prediction model using a recurrent neural network algorithm for the Seoul metropolitan area, Republic of Korea, Atmos. Environ. 245 (2021) 118021.
[6] J. Chen, J. Lu, J.C. Avise, J.A. DaMassa, M.J. Kleeman, A.P. Kaduwela, Seasonal modeling of pm2. 5 in California's San Joaquin Valley, Atmos. Environ. 92 (2014) 182–190.
[7] Y. Chen, J. An, et al., A novel prediction model of pm2. 5 mass concentration based on back propagation neural network algorithm, J. Intell. Fuzzy Syst. 37 (2019) 3175–3183.
[8] G. Corani, M. Scanagatta, Air pollution prediction via multi-label classification, Environ. Model. Softw. 80 (2016) 259–264.
[9] M. Cordano, I.H. Frieze, Pollution reduction preferences of US environmental managers: applying Ajzen's theory of planned behavior, Acad. Manag. J. 43 (2000) 627–641.
[10] P. Duffy, M.E. Cassida, C. Brion, D. Chong, Assessment of Gaussian-weighted angular resolution functions in the comparison of quantum-mechanically calculated electron momentum distributions with experiment, Chem. Phys. 159 (1992) 347–363.
[11] R. Feng, H. Gao, K. Luo, J.r. Fan, Analysis and accurate prediction of ambient pm2. 5 in China using multi-layer perceptron, Atmos. Environ. 232 (2020) 117534.
[12] X. Feng, Q. Li, Y. Zhu, J. Hou, L. Jin, J. Wang, Artificial neural networks forecasting of pm2. 5 pollution using air mass trajectory based geographic model and wavelet transformation, Atmos. Environ. 107 (2015) 118–128.
[13] I.H. Fong, T. Li, S. Fong, R.K. Wong, A.J. Tallón-Ballesteros, Predicting concentration levels of air pollutants by transfer learning and recurrent neural network, Knowl.-Based Syst. 192 (2020) 105622.
[14] M.L. González-Martínez, L. Bonnet, P. Larrégaray, J-C. Rayez, Classical treatment of molecular collisions: striking improvement of the description of recoil energy distributions using Gaussian weighted trajectories, J. Chem. Phys. 126 (4) (2007) 041102, American Institute of Physics.
[15] K. Gu, J. Qiao, X. Li, Highly efficient picture-based prediction of pm2. 5 concentration, IEEE Trans. Ind. Electron. 66 (2018) 3176–3184.
[16] Y. Hao, Y.M. Liu, The influential factors of urban pm2. 5 concentrations in China: a spatial econometric analysis, J. Clean. Prod. 112 (2016) 1443–1453.

[17] M. Hossain, B. Rekabdar, S.J. Louis, S. Dascalu, Forecasting the weather of Nevada: a deep learning approach, in: 2015 International Joint Conference on Neural Networks (IJCNN), IEEE, 2015, pp. 1–6.
[18] R. Karim, T.H. Rafi, An automated lstm-based air pollutant concentration estimation of Dhaka city, Bangladesh, Int. J. Eng. Inf. Syst. 4 (2020) 88–101.
[19] H.Y. Kim, C.H. Won, Forecasting the volatility of stock price index: a hybrid model integrating lstm with multiple garch-type models, Expert Syst. Appl. 103 (2018) 25–37.
[20] M. Kolehmainen, H. Martikainen, J. Ruuskanen, Neural networks and periodic components used in air quality forecasting, Atmos. Environ. 35 (2001) 815–825.
[21] W. Kong, Z.Y. Dong, Y. Jia, D.J. Hill, Y. Xu, Y. Zhang, Short-term residential load forecasting based on lstm recurrent neural network, IEEE Trans. Smart Grid 10 (2017) 841–851.
[22] A. Kurt, A.B. Oktay, Forecasting air pollutant indicator levels with geographic models 3 days in advance using neural networks, Expert Syst. Appl. 37 (2010) 7986–7992.
[23] Y. La Su, W.W. Liu, et al., Research on the lstm Mongolian and Chinese machine translation based on morpheme encoding, Neural Comput. Appl. 32 (2020) 41–49.
[24] V.D. Le, T.C. Bui, S.K. Cha, Spatiotemporal deep learning model for citywide air pollution interpolation and prediction, in: 2020 IEEE International Conference on Big Data and Smart Computing (BigComp), IEEE, 2020, pp. 55–62.
[25] A. Lee, A. Szpiro, S. Kim, L. Sheppard, Impact of preferential sampling on exposure prediction and health effect inference in the context of air pollution epidemiology, Environmetrics 26 (2015) 255–267.
[26] X. Li, L. Peng, X. Yao, S. Cui, Y. Hu, C. You, T. Chi, Long short-term memory neural network for air pollutant concentration predictions: method development and evaluation, Environ. Pollut. 231 (2017) 997–1004.
[27] Z. Lin, M. Li, Z. Zheng, Y. Cheng, C. Yuan, Self-attention convlstm for spatiotemporal prediction, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2020, pp. 11531–11538.
[28] H. Maleki, A. Sorooshian, G. Goudarzi, Z. Baboli, Y.T. Birgani, M. Rahmati, Air pollution prediction by using an artificial neural network model, Clean Technol. Environ. Policy 21 (2019) 1341–1352.
[29] H. Mayer, Air pollution in cities, Atmos. Environ. 33 (1999) 4029–4037.
[30] Galen McKinley, Miriam Zuk, Morten Höjer, Montserrat Avalos, Isabel González, Rodolfo Iniestra, Israel Laguna, Miguel A. Martínez, Patricia Osnaya, Luz M. Reynales, Raydel Valdés, Julia Martínez, Quantification of local and global benefits from air pollution control in Mexico City, Environ. Sci. Technol. 39 (7) (2005) 1954–1961, https://doi.org/10.1021/es035183e.
[31] H. Meng, K. Wang, Y. Gao, Y. Jin, X. Ma, J. Tian, Adaptive Gaussian weighted Laplace prior regularization enables accurate morphological reconstruction in fluorescence molecular tomography, IEEE Trans. Med. Imaging 38 (2019) 2726–2734.
[32] M. Mokhtari, M. Miri, A. Mohammadi, H. Khorsandi, Y. Hajizadeh, A. Abdolahnejad, Assessment of air quality index and health impact of pm10, pm2. 5 and so2 in Yazd, Iran, J. Maz. Univ. Med. Sci. 25 (2015) 14–23.
[33] S. Park, M. Kim, M. Kim, H.G. Namgung, K.T. Kim, K.H. Cho, S.B. Kwon, Predicting pm10 concentration in Seoul metropolitan subway stations using artificial neural network (ann), J. Hazard. Mater. 341 (2018) 75–82.
[34] K. Qadeer, W.U. Rehman, A.M. Sheri, I. Park, H.K. Kim, M. Jeon, A long short-term memory (lstm) network for hourly estimation of pm2. 5 concentration in two cities of South Korea, Appl. Sci. 10 (2020) 3984.
[35] Y. Qi, Q. Li, H. Karimian, D. Liu, A hybrid model for spatiotemporal forecasting of pm2. 5 based on graph convolutional neural network and long short-term memory, Sci. Total Environ. 664 (2019) 1–10.
[36] D. Qin, J. Yu, G. Zou, R. Yong, Q. Zhao, B. Zhang, A novel combined prediction scheme based on cnn and lstm for urban pm 2.5 concentration, IEEE Access 7 (2019) 20050–20059.
[37] A.G. Russell, K.F. McCue, G.R. Cass, Mathematical modeling of the formation of nitrogen-containing air pollutants. 1. Evaluation of an Eulerian photochemical model, Environ. Sci. Technol. 22 (1988) 263–271.
[38] P.E. Saide, G.R. Carmichael, S.N. Spak, L. Gallardo, A.E. Osses, M.A. Mena-Carrasco, M. Pagowski, Forecasting urban pm10 and pm2. 5 pollution episodes in very stable nocturnal conditions and complex terrain using wrf–chem co tracer model, Atmos. Environ. 45 (2011) 2769–2780.
[39] A. Suleiman, M. Tight, A. Quinn, Applying machine learning methods in managing urban concentrations of traffic-related particulate matter (pm10 and pm2. 5), Atmos. Pollut. Res. 10 (2019) 134–144.
[40] W. Sun, H. Zhang, A. Palazoglu, A. Singh, W. Zhang, S. Liu, Prediction of 24-hour-average pm2. 5 concentrations using a hidden Markov model with different emission distributions in Northern California, Sci. Total Environ. 443 (2013) 93–103.
[41] M. Sundermeyer, H. Ney, R. Schlüter, From feedforward to recurrent lstm neural networks for language modeling, IEEE/ACM Trans. Audio Speech Lang. Process. 23 (2015) 517–529.
[42] J. Tian, D. Chen, A semi-empirical model for predicting hourly ground-level fine particulate matter (pm2. 5) concentration in southern Ontario from satellite remote sensing and ground-based meteorological measurements, Remote Sens. Environ. 114 (2010) 221–229.
[43] J. Wang, S.A. Christopher, Intercomparison between satellite-derived aerosol optical thickness and pm2. 5 mass: implications for air quality studies, Geophys. Res. Lett. 30 (2003).
[44] J.Q. Wang, Y. Du, J. Wang, Lstm based long-term energy consumption prediction with periodicity, Energy 197 (2020) 117197.
[45] Z. Wang, T. Maeda, M. Hayashi, L.F. Hsiao, K.Y. Liu, A nested air quality prediction modeling system for urban and regional scales: application for high-ozone episode in Taiwan, Water Air Soil Pollut. 130 (2001) 391–396.
[46] X. Xu, M. Yoneda, Multitask air-quality prediction based on lstm-autoencoder model, IEEE Trans. Cybern. (2019).
[47] Z. Xu, Y. Lv, Att-convlstm: Pm 2.5 prediction model and application, in: The International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery, Springer, 2019, pp. 30–40.
[48] F. Xue, H. Ji, W. Zhang, Y. Cao, Attention-based spatial–temporal hierarchical convlstm network for action recognition in videos, IET Comput. Vis. 13 (2019) 708–718.
[49] R. Yan, J. Liao, J. Yang, W. Sun, M. Nong, F. Li, Multi-hour and multi-site air quality index forecasting in Beijing using cnn, lstm, cnn-lstm, and spatiotemporal clustering, Expert Syst. Appl. 169 (2021) 114513.
[50] J. Yang, M. Hu, Filling the missing data gaps of daily modis aod using spatiotemporal interpolation, Sci. Total Environ. 633 (2018) 677–683.
[51] W. Yang, M. Deng, F. Xu, H. Wang, Prediction of hourly pm2. 5 using a space-time support vector regression model, Atmos. Environ. 181 (2018) 12–19.
[52] J. Yi, Z. Wen, J. Tao, H. Ni, B. Liu, Ctc regularized model adaptation for improving lstm rnn based multi-accent Mandarin speech recognition, J. Signal Process. Syst. 90 (2018) 985–997.
[53] X. Yi, J. Zhang, Z. Wang, T. Li, Y. Zheng, Deep distributed fusion network for air quality prediction, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 965–973.
[54] X. Yuan, S. Qi, Y. Wang, Stacked enhanced auto-encoder for data-driven soft sensing of quality variable, IEEE Trans. Instrum. Meas. 69 (2020) 7953–7961.
[55] M. Zamani Joharestani, C. Cao, X. Ni, B. Bashir, S. Talebiesfandarani, Pm2. 5 prediction based on random forest, xgboost, and deep learning using multisource remote sensing data, Atmosphere 10 (2019) 373.
[56] B. Zhang, H. Zhang, G. Zhao, J. Lian, Constructing a pm2. 5 concentration prediction model by combining auto-encoder with bi-lstm neural networks, Environ. Model. Softw. 124 (2020) 104600.
[57] B. Zhang, G. Zou, D. Qin, Y. Lu, Y. Jin, H. Wang, A novel encoder-decoder model based on read-first lstm for air pollutant prediction, Sci. Total Environ. 765 (2021) 144507.
[58] J.Y. Zhu, C. Sun, V.O. Li, An extended spatio-temporal granger causality model for air quality estimation with heterogeneous urban big data, IEEE Trans. Big Data 3 (2017) 307–319.
[59] J.Y. Zhu, C. Zhang, H. Zhang, S. Zhi, V.O. Li, J. Han, Y. Zheng, pg-Causality: identifying spatiotemporal causal pathways for air pollutants with urban big data, IEEE Trans. Big Data 4 (2017) 571–585.
[60] Y.Y. Zhu, Y.X. Gao, B. Liu, X.Y. Wang, L.L. Zhu, R. Xu, W. Wang, J.N. Ding, J.J. Li, X.L. Duan, Concentration characteristics and assessment of model-predicted results of pm2. 5 in the Beijing-Tianjin-Hebei region in autumn and winter, Huanjing Kexue 40 (2019) 5191–5201.