TU Delft

MSc. Thesis Report

PM2.5 concentration prediction and early warning system of extreme conditions based on the LSTM

Siyu Guan

Supervisors:
Prof. dr. ir. H. X. Lin
Dr. J. J. Cai

August 15, 2018

MSc. Thesis Report

PM2.5 concentration prediction and early warning system of extreme conditions based on the LSTM

by

Siyu Guan

Student number: 4623843
Faculty: EEMCS
Master Program: Applied Mathematics
Thesis committee: Prof. dr. ir. H. X. Lin
Dr. J. J. Cai
Dr. J.W. van der Woude

TU Delft

Abstract

This thesis project developed an alternative PM2.5 concentration prediction model and early warning system of extreme air pollution based on the long short-term memory (LSTM) and achieved satisfying performance. To research more deeply, we divided the task into two parts. The first task was predicting the PM2.5 concentration of next 24 hours and another one was building early warning system of extreme air pollution of next 12 hours.

To solve the first task, we started from the 1-hour prediction problem, that was predicting PM2.5 of next hour based on the last hours' data. We did parameter optimization to derive the best network architecture and we got a RMSE of 19.7863. We then successfully built 24-hour prediction model that was predicting PM2.5 concentration of next 24 hours according to the optimal 1-hour prediction model. The proposed 24-hour prediction model exhibited satisfactory performance, including the 13-24 h prediction task which is predicting the mean PM2.5 concentration among next 13-24 hours (RMSE=49.41).

Although we got a satisfying RMSE for the PM2.5 prediction problem, we didn't get accurate prediction for extreme conditions and that's why we continued to focus on the second task. We regarded the highest PM2.5 value among 12 hours as the extreme air pollution of this period and we divided the warning level into 4 parts. Then we built the early warning system based on the LSTM to predict the warning level of highest PM2.5 value of next 12 hours. As indicated by the ACC and AUC, our LSTM model achieved sound performance (ACC=86.7%, AUC=0.837).

To improve the prediction performance, we focused on several model optimization techniques for the 1-hour prediction model and each technique has effectively improved the accuracy. Moreover, we combined these optimization methods together, which leaded to the lowest RMSE of 14.1937. The combined optimization method performed better than any single optimization method, which suggested that we can use some effective optimization methods together to improve the prediction accuracy of LSTM model. In addition, we also compared our model with the random forest (RF) model and the comparison result proved that LSTM network worked better for both tasks.

Content

1. Introduction

1.1 Research problem

With the growth of economy, air pollution is gradually increasing in many developing areas which leads to the change of climate and threatens human survival. Air pollutants have significant influence on the natural environment, and could also harm vegetation and monuments[1]. In addition to ordinary air pollution, the phenomenon of extreme air pollution (in our thesis project, we define extreme air pollution as the air condition with a very high value of PM2.5) has also become more frequent in recent years. Extreme air pollution highly threatens people health and economic development. Thus we need to build a reliable PM2.5 predicting system and to set up an early warning system of the extreme air pollution.

Accuracy and efficiency are especially important in the prediction of PM2.5 and there has been a lot of improvements in research on air pollution prediction. However, existing global or national PM2.5 forecast models are not able to provide sufficient details and accuracy for local areas, often a city of size 10km by 10km is only one grid point in the model. What's more, the accurate prediction on the extreme air pollution is still a research challenge and worth studying[2-3].

Prediction methods can be applied to many cases [4-6] and there are mainly two type of methods: numerical models based on Partial Differential Equations (PDEs) and statistical models[7]. Apart from traditional prediction methods, artificial intelligence (AI) has been used in forecasting problems and achieves outstanding performance[8]. AI and deep learning algorithms can process high-dimensional information cross a variety of disciplines. Applying AI techniques can effectively improve prediction skill for air pollution and extreme air pollution[2-3,8]. Besides, deep learning provides many outstanding performance, like easily dealing with extensive data and increasing the chance of reaching accuracy with the greater degree of precision.

Recurrent neural network (RNN) and long short- term memory (LSTM) models are two kinds of representative algorithms in AI methods. However, in the training process of RNN, the gradient disappearing phenomenon appears which greatly restricts the ability of model (see further explanation in section 1.2). A special RNN model, LSTM is developed to overcome

the shortcomings of RNN . LSTM and traditional RNN exhibit outstanding results for time series forecasting cases[9], thus are especially suitable for PM2.5 concentration prediction tasks. We will build a prediction model for PM2.5 concentration and build an early warning system of extreme air pollution based on LSTM model in our thesis.

1.2 Previous research

In recent years, the research on air pollution forecasting methods has achieved some progress. Numerical models based on PDEs simulate weather patterns and statistical methods give prediction mainly based on the corresponding historical data [10].

For the numerical models based on PDEs, the Chemical Transport Model (CTM)[11], CMAQ [12] and WRFChem[13] has been applied to predict the air pollution. Based on the CTM model, a lot of software were developed to make prediction for gas diffusion, climate change and so on. In order to simulate the diffusion and chemical change of aerial contaminant, the LOTOS-EUROS CTM is established in Holland and is widely applied in Europe[14]. The CTM method can be used with a small amount of training samples and it is able to predict the density of contaminant for the area without observatories. However, the prediction precision is strongly depending on the information quality and magnitude of the problem. Although these models give clear simulation of dispersion, advanced physical-chemistry knowledge is required for most of these models for the sake of reliability[15].

For statistical methodology, the random forest model (RF)[16], the multi-linear regression (MLR)[17], the support vector regression (SVR)[18], neural network (NN) [19] and hybrid methods[20] are broadly utilized for predicting the extent of contaminant. In these methods, NN is broadly employed in current research and typically produces good results. NN can deal with nonlinear problems easily and has high efficiency and flexibility. Moreover, many variants of NN emerged which significantly broaden the range of application. Representative methods involve the convolutional neural network (CNN)[21], back propagation neural network (BPNN)[22], multilayer perceptron (MLP)[23], general regression neural network (GRNN) [24] and recurrent neural network (RNN)[25].

Conventional NN has a simple connection structure where each processing vector is independent from one another. However, to predict the air quality in a short period ahead, the current status of air quality is very likely to be informative. RNN can deal with the same problem for each unit within the recurrent structure and past calculation will influence the next

operation[26]. Combining with the variational trend of air pollution, RNN is especially suitable for processing time series data and discovering potential principle[27]. Previous researches also show the applicability of RNN on pollution forecast[28,29].

However, in the training process of RNN, the calculated gradients which are important for the parameter updates always exponentially increase or decrease. Therefore, the ability of RNN for simulating long-term data is restricted, which affects the performance of RNN for time series prediction. Unlike traditional RNN, as a special variant of RNN, LSTM has capacity of modelling long-range data and doesn't have the gradients decreasing problems[30]. This advantage is particularly essential for modelling the dispersion of aerial contaminant.

Recently, there are many successful application of LSTM model including stock pricing forecasting[30], photovoltaic power forecasting[31], earthquake prediction[32], etc. For the prediction of air quality, Sak[33] utilized LSTM method to predict the hazards of pollutants. However, they didn't predict the specific density of air pollutants and only focused on the forecasting of hazard level. Besides, they gave forecasts only for single observation station and didn't analyse the correlation among different areas.

To overcome these drawbacks and improve the prediction performance, Xiang Li[34] proposed a LSTM based method which successfully predicts the PM2.5 concentrations for next 24h. In order to derive an accurate prediction, they divided the next 24 h into 6 time lags (1h,2h,3h,4-6h,7-12h,13-24h). They built the prediction model separately and forecast the average density of pollutants for each period. The proposed model exhibited a root mean square error (RMSE) of 12.60 for 1h prediction tasks and they got a RMSE of 41.94 for 13-24h prediction tasks. Compared with their experiments on traditional method (RMSE of SVR is 22.04, RMSE of ARMA is 24.40) and traditional neural network (RMSE is 16.19), the result of LSTM is satisfactory.

However, their research did not consider three issues: 1) the prediction performance on the extreme air pollution. Thus, in the evaluation of the model they also neglected the prediction error on the extreme air pollution (high peak value of PM2.5 concentration which are most relevant because they have large impact on human life) 2) in the construction of LSTM network architecture, they used the default parameters for simplicity and built model with fixed network structure without a scientific explanation and validation. 3) optimization techniques to improve the model performance which is particularly important because of the restricted amount of historical datasets available for the modelling.

1.3 Thesis objective

Based on the literature review and discussions in section 1.2, this thesis project will develop an alternative methodology for PM2.5 concentration prediction and in particular focusing on extreme air pollution by designing an LSTM based system. It should be able to learn inherent principle from a large number of variables and also performs well for extreme air pollution. More specifically, we will focus on the following research targets:

(1) build a basic air quality prediction model based on the LSTM
(2) investigate the influence of model structure and data structure on the prediction performance including the basic and extreme air pollution conditions
(3) compare the performance of LSTM with other method (Random Forest)
(4) optimise the model by optimization techniques including neural network dropout and decreasing learning rate
(5) finally, build an air pollution prediction and early-warning system of extreme condition based on the LSTM

2. Introduction to RNN and LSTM

In this chapter, we will introduce the theoretical basis of RNN and LSTM which are important foundation to build the PM2.5 concentration prediction model. We will start from the network structure and the recurrent process in the RNN. After analysing the advantages and disadvantages of RNN, we will introduce the LSTM model and discuss the difference and the improvements compared with RNN. Then we will introduce the whole process from the input vector x_t to the output vector h_t when a single LSTM unit gets the input vector x_t at timestep t. Finally, we will show how the back propagation through time (BPTT) and gradient descent (GD) are used to update parameters and give better predictions.

2.1 Nomenclatures

Table 1 is the description of some important symbols which are involved in this thesis project.

Table 1. Description of symbols

Symbol	Description
$x_t \in R^d$	Input of a LSTM
$h_t \in R^h$	Output of a LSTM
σ	$\sigma(x) = \frac{1}{1+e^{-x}}$
tanh	$\tanh(x) = 2\sigma(2x) - 1 = \frac{2}{1+e^{-2x}} - 1$
$f_t \in R^h$	Forget gate's activation vector
$W_f \in R^{h\times(h+d)}$	Weight matrix of forget gate
$b_f \in R^h$	Bias of forget gate
$i_t \in R^h$	Input gate's activation vector
$W_i \in R^{h\times(h+d)}$	Weight matrix of input gate
$b_i \in R^h$	Bias of input gate
$\tilde{c}_t \in R^h$	Candidate of cell state $c_t \in R^h$
γ	Learning rate

$W_C \in R^{h\times(h+d)}$	Weight matrix of cell state
$b_C \in R^h$	Bias of cell state
$c_t \in R^h$	Cell state vector
$o_t \in R^h$	Output gate's activation vector
$W_o \in R^{h\times(h+d)}$	Weight matrix of output gate
$b_o \in R^h$	Bias vector of output gate
$*$	Multiplication by elements

2.2 RNN

As mentioned above, we first introduce the model structure and properties of RNN. The idea behind RNN is to utilize the information within sequences. For a conventional NN, it is assumed that all input vectors are independent and similar assumptions are applicable for outputs as well. However, for many real-world problems, these assumptions are unrealistic. We better know the previous air pollution concentration if we need to forecast the air pollutant concentration at the next timestep. RNN is called recurrent because the same task is performed for each element of the series and the output depends on past calculations.

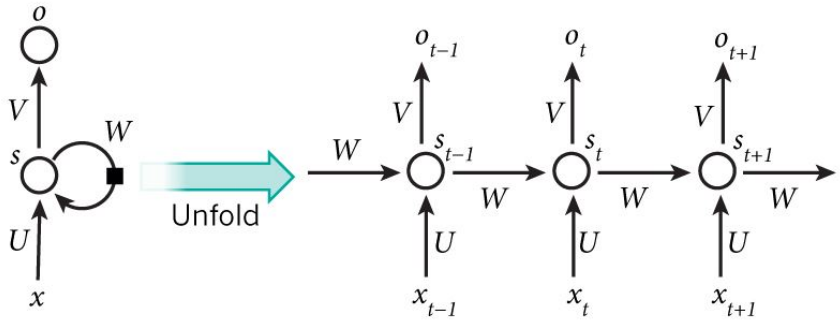

Fig. 1. RNN and its unfolding form

Fig. 1 presents the structure of RNN and its unrolled form. The computations happened per timestep are as followings[31]:

1) x_t represents the input at timestep t. In the PM2.5 concentration prediction tasks, x_t should provide the information like PM2.5 value and wind direction at time t which we think is likely to be helpful to forecast the PM2.5 values at time t +1.
2) s_t refers to the hidden state at timestep t, which is designed to 'memory' the information. In timestep t, its hidden state s_t is calculated according to the hidden state at timestep

t-1 (s_{t-1}) and the input at timestep t (x_t). That is, $s_t = f(Ux_t + Ws_{t-1})$. In general, function f mentioned here is a nonlinear function such as *sigmoid* or *ReLU*. As for the first hidden state s_{-1}, it is usually set as zero for initialization.

3) o_t corresponds to the output at timestep t. In the task of PM2.5 concentration prediction, it would be the predicted PM2.5 concentration at time t. $O_t = g(Vs_t)$. Typically, the function g is a nonlinear function such as *sigmoid* or *ReLU*.

Through unrolling, it is shown that a RNN can be used to deal with time series. Another perspective of RNN is that they are capable to extract information obtained so far by means of "memory". In the PM2.5 concentration prediction tasks, this "memory" can be regarded as the information about the past changes in PM2.5 and other data we input to the model like wind direction or temperature. Theoretically, RNN is able to "memory" information in arbitrarily long series, but practically the length is restricted to just a few timesteps.

2.3 LSTM

LSTM is a branch of RNN and is specially developed to overcome the gradient decreasing problem that RNN face. Compared with RNN, LSTM also has the chain structure but the module inside is replaced with a different setting. Since RNN is sensitive for short term memory and insensitive for long term memory, LSTM adds a special cell state C to store the long term memory to address the problems.

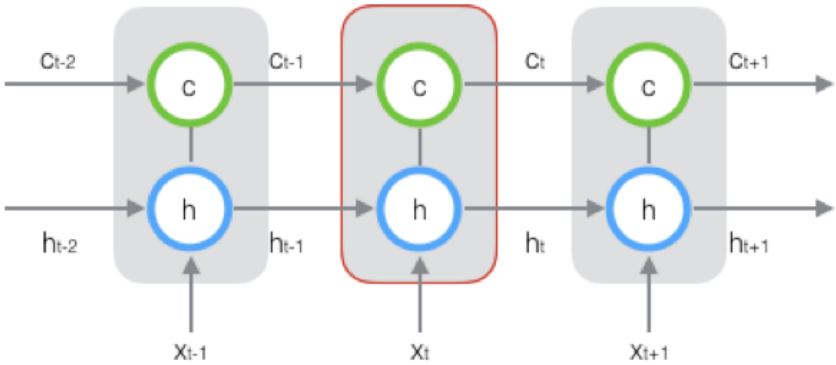

Fig. 2. A sketch for the LSTM structure. Hidden layer h and cell state c are included in the red solid box which is called a LSTM unit.

Fig. 2 is a sketch for the LSTM structure and the red solid box in the figure is called a LSTM unit. We can see that at time t there are 3 input vectors (x_t, h_{t-1} *and* c_{t-1}) and 2 output vectors (h_t *and* c_t) for the LSTM unit. Within the LSTM unit, the green cycle is called the cell state

which is the key part for this unit as well as for the whole LSTM structure. Gates are a special way for the LSTM unit to optionally let information through. Fig. 2 is just a sketch for the LSTM and the network structure within a single LSTM unit is not as simple as it looks in the figure. In fact, there is a complex network structure within the LSTM unit and it uses three type of gates to protect and control itself.

In the following part, we will introduce the network structure within a single LSTM unit which are important theoretical basis for the LSTM model we built in next chapter. Moreover, in the training process of the general LSTM networks, the Back-propagation Through Time (BPTT) algorithm and Gradient Descent (GD) are used to update weights. Although BPTT and GD are used in the whole LSTM model, the principle and process are same for each single part of the LSTM network. For this reason, we will just start from the forward propagation for a single LSTM unit to introduce the LSTM unit structure and the principle of BPTT. After that we will discuss the back propagation of BPTT within a single LSTM unit and the GD method for parameter updates.

2.3.1 Forward propagation

BPTT involves forward propagation and back propagation and this algorithm is widely used in many type of neural networks. The forward propagation can be considered as the propagation of input data through the network and we will have an output after this process. Then we will compare the output with the value we want and calculate the error. To reduce this error, we need give the feedback to the network and change parameters of the network (parameters are shown in Table 2 of section 2.3.2). That is why we use back propagation to propagate the error and use GD to find the best way to change parameters to reduce the error. By repeating this forward and back propagation, we can improve the prediction performance step by step. This repeating process is usually called the training process of the model while the dataset used for modelling are training dataset.

We now start from the forward propagation to introduce the training process for a single LSTM unit. We will explain how the LSTM works with the special cell state and three gates when it receives the input data $x = (x_1, ..., x_T)$ at time t. Generally speaking, LSTM uses 2 gates to control the contents of cell state C_t. One is the forget gate that is designed to decide how much of C_{t-1} is retained for C_t. The other is the input gate, which determines how much of the x_t is saved to C_t. Then LSTM uses 1 gate that is output gate to control how many information of C_t is output to h_t.

(1) Forget gate

Firstly, the LSTM determines what information will be forgotten. In the PM2.5 concentration prediction tasks, not all the data about the past are important and we need to forget some unimportant information, which is decided by a sigmoid layer, namely the forget gate layer.

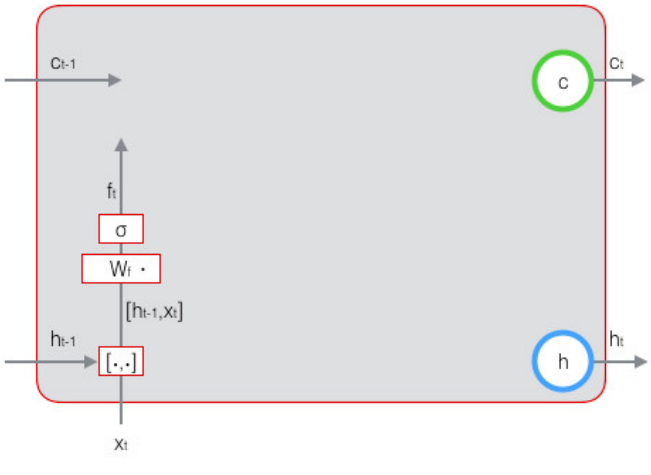

Fig. 3. Forget gate in LSTM

In Fig. 3, [.,.] represents the vector embedding operation that combines 2 vectors to a single vector. And the corresponding formula of forget gate is shown in equation (1).

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ &= \sigma([W_f] \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix} + b_f) \\ &= \sigma([W_{fh}\ W_{fx}] \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix} + b_f) \\ &= \sigma\left(W_{fh} h_{t-1} + W_{fx} x_t + b_f\right) \end{aligned} \tag{1}$$

In equation (1), $h_{t-1} \in R^h$ is the output of a LSTM at timestep t -1 and $x_t \in R^d$ is the air condition received by the LSTM at timestep t. $W_f \in R^{h\times(h+d)}$ is the weight matrix of forget gate and $b_f \in R^h$ is the bias of forget gate. Moreover, to make it easier to understand, we can think that W_f is embedded by two weight matrix, which are $W_{fh} \in R^{h\times h}$ and $W_{fx} \in R^{h\times d}$. We do this representation because we will use the W_{fh}, W_{fx} in the following part of parameter updating. Besides, σ refers to the sigmoid function

$$\sigma(x)=\frac{1}{1+e^{-x}} \tag{2}$$

and $f_t \in R^h$ is the forget gate's activation vector.

(2) Input gate and candidate cell state $\tilde{c}_t$

Secondly, the LSTM determines what new information will be stored. If we want to forecast the value of PM2.5 including the extreme air pollution, we have to store the important information about the past. This is achieved by two parts. First, the input gate layer determines which value will be updated and is shown in equation (3),

$$\begin{aligned} i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ &= \sigma\left(W_{ih} h_{t-1} + W_{ix} x_t + b_i\right) \end{aligned} \tag{3}$$

in which $i_t \in R^h$, $W_i \in R^{h\times(h+d)}$ and $b_i \in R^h$ are activation vector, weight matrix and bias in the input gate respectively. Then we calculate the candidate cell state $\tilde{c}_t \in R^h$ which is calculated by the previous time output and current input and describes current input status.

$$\begin{aligned} \tilde{C}_t &= \tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right) \\ &= \tanh\left(W_{Ch} h_{t-1} + W_{Cx} x_t + b_C\right) \end{aligned} \tag{4}$$

Fig. 4 describes the calculation of candidate cell state and input gate.

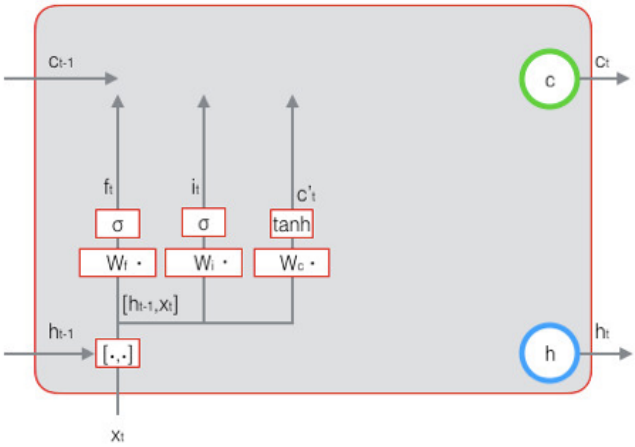

Fig. 4. Input gate and candidate cell state

(3) Cell state C_t

Now we calculate the cell state C_t as equation (6) in which * represents multiplication by elements as equation (5).

$$a*b=\begin{bmatrix}a_1\\a_2\\a_3\\\cdots\\a_n\end{bmatrix}*\begin{bmatrix}b_1\\b_2\\b_3\\\cdots\\b_n\end{bmatrix}=\begin{bmatrix}a_1b_1\\a_2b_2\\a_3b_3\\\cdots\\a_nb_n\end{bmatrix} \tag{5}$$

We first multiply the old state by f_t to forget the information we need to forget. Continually, w $i_t*\tilde{C}_t$ is added, which shows how many information we wanted to learn from candidate cell state $\tilde{C}_t$.

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

Fig. 5 shows the calculation of C_t.

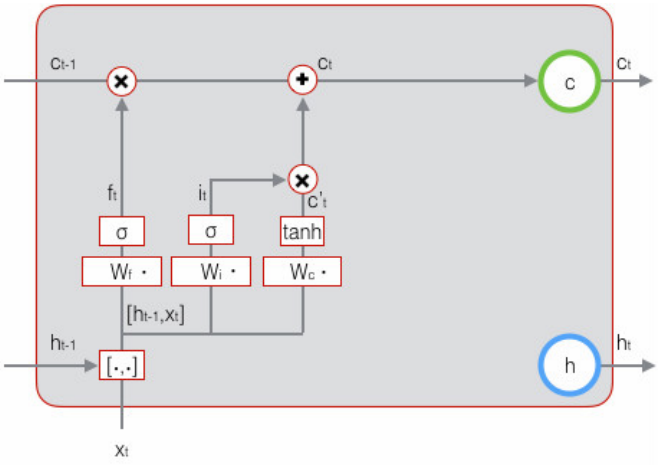

Fig. 5. Calculation of C_t

By equation (6), we combine the current candidate memory $\tilde{C}_t$ with the long term memory C_{t-1} together and have a new cell state C_t. Through the control of forget gate and input gate, C_t can save the information about long time ago and avoid the entering of insignificant information to the current memory.

(4) Output gate

In the end, the LSTM determines what will be outputted. Equation (7) shows the calculation of the output gate and $o_t \in R^h$ is named the activation of the output gate.

$$\begin{aligned}o_t&=\sigma(W_o\cdot[h_{t-1},x_t]+b_o)\\&=\sigma\left(W_{oh}h_{t-1}+W_{ox}x_t+b_o\right)\end{aligned} \tag{7}$$

The final output $h_t \in R^h$ is obtained by combining the output gate's activation vector o_t and the cell state c_t .

$$h_t = o_t * \tanh(c_t) \tag{8}$$

Equation (8) shows the calculation of the final output vector of the LSTM unit. And Fig. 6 shows the calculation of the final output.

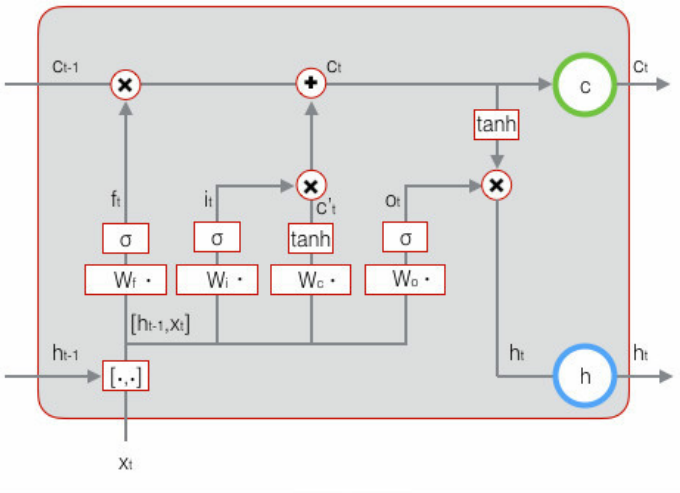

Fig. 6. Output gate and final output h_t

We can also see the whole forward calculation process in Fig. 6. Now we have the value of each neural units of LSTM (f_t, i_t, c_t, o_t *and* h_t) and we need to do a back propagation to know the errors of each neural units which provides essential information for parameters updating.

2.3.2 Back propagation

There are 8 type of parameters need to be trained and updated, which are given in Table 2.

Table 2. Parameters need to be trained in LSTM

Parameter	Description	Parameter	Description
$W_f = [W_{fh}\ W_{fx}]$	weight matrix of forget gate	b_f	bias of forget gate
$W_i = [W_{ih}\ W_{ix}]$	weight matrix of input gate	b_i	bias of input gate
$W_c = [W_{ch}\ W_{cx}]$	weight matrix of cell state	b_c	bias of cell state
$W_o = [W_{oh}\ W_{ox}]$	weight matrix of output gate	b_o	bias of output gate

During the back propagation, we will make a comparison between the output values and the expected results and then evaluate their errors. Then the derivatives of error function about the network weights will be calculated and we will update the weights to minimize the error.

(1) Back propagation of error term through time

At timestep *t*, the output of a LSTM is h_t. We define the error term δ_t at time *t* as

$$\delta_t \overset{def}{=} \frac{\partial E}{\partial h_t} \tag{9}$$

In which *E* is loss function. In PM2.5 prediction tasks, loss function is normally taken as mean square error (MSE) to minimize the expected loss. In our thesis, we take MSE as the loss function.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - y_i^*\right)^2 \tag{10}$$

In equation (10), y_i^* is the real PM2.5 value and y_i is the calculated PM2.5 value and *n* is the number of test samples. Back propagation of error term through time means we want to calculate the error term of previous time δ_{t-1}. Through mathematical derivation, we can prove that δ_{t-1} can be calculated by equation (11).

$$\delta_{t-1} = \delta_{f,t}^T W_{fh} + \delta_{i,t}^T W_{ih} + \delta_{c,t}^T W_{ch} + \delta_{o,t}^T W_{oh} \tag{11}$$

Equation (12) to (15) are the definition of $\delta_{f,t}^T, \delta_{i,t}^T, \delta_{c,t}^T$ *and* $\delta_{o,t}^T$.

$$\delta_{f,t}^T = \delta_t^T * o_t * \left(1 - \tanh\left(c_t\right)^2\right) * c_{t-1} * f_t * \left(1 - f_t\right) \tag{12}$$

$$\delta_{i,t}^T = \delta_t^T * o_t * \left(1 - \tanh\left(c_t\right)^2\right) * \tilde{c}_t * i_{tt} * \left(1 - i_t\right) \tag{13}$$

$$\delta_{c,t}^T = \delta_t^T * o_t * \left(1 - \tanh\left(c_t\right)^2\right) * i_{tt} * \left(1 - \tilde{c}_t^2\right) \tag{14}$$

$$\delta_{o,t}^T = \delta_t^T * \tanh\left(c_t\right) * o_t * \left(1 - o_t\right) \tag{15}$$

Equation (11) (12) are the formulas to achieve back propagation of error term through time, and we can calculate the error term at time *k*.

$$\delta_k^T = \prod_{j=k}^{t-1}\left(\delta_{f,j}^T W_{fh} + \delta_{i,j}^T W_{ih} + \delta_{c,j}^T W_{ch} + \delta_{o,j}^T W_{oh}\right) \tag{16}$$

(2) Back propagation of error term through layer

Assume current layer is the l th layer, then we can prove that the error term of l-1 layer can be calculated by equation (17).

$$\delta_t^{l-1} = \left(\delta_{f,t}^T W_{fx} + \delta_{i,t}^T W_{ix} + \delta_{c,t}^T W_{cx} + \delta_{o,t}^T W_{ox}\right) * f^{l-1}\left(net_t^{l-1}\right) \tag{17}$$

In equation (17), f^{l-1} represents the activation function of layer l-1 and net_t^{l-1} is the weighted sum of input to the layer l-1.

(3) Calculation of gradient

Based on the error term, the gradient can be calculated with equation (18) to (25).

$$\frac{\partial E}{\partial W_{fh}} = \sum_{j=1}^{t} \delta_{f,j} h_{j-1}^T \tag{18}$$

$$\frac{\partial E}{\partial W_{ih}} = \sum_{j=1}^{t} \delta_{i,j} h_{j-1}^T \tag{19}$$

$$\frac{\partial E}{\partial W_{ch}} = \sum_{j=1}^{t} \delta_{c,j} h_{j-1}^T \tag{20}$$

$$\frac{\partial E}{\partial W_{oh}} = \sum_{j=1}^{t} \delta_{o,j} h_{j-1}^T \tag{21}$$

$$\frac{\partial E}{\partial b_f} = \sum_{j=1}^{t} \delta_{f,j} \tag{22}$$

$$\frac{\partial E}{\partial b_i} = \sum_{j=1}^{t} \delta_{i,j} \tag{23}$$

$$\frac{\partial E}{\partial b_c} = \sum_{j=1}^{t} \delta_{c,j} \tag{24}$$

$$\frac{\partial E}{\partial b_c} = \sum_{j=1}^{t} \delta_{c,j} \tag{25}$$

After we have the gradient of the weight, we will introduce how to use gradient descent to adjust the weights to minimize the error.

2.3.3 Gradient Descent (GD)

We want to minimize the loss function to have a better prediction and we already calculate the gradient, now we use Gradient Descent (GD) to do the parameter updates. GD as an optimization method is usually applied to search for the coefficients that AI algorithms need to

determine. The target of GD is to find the parameters which can minimize the objective function on the train data. We will first introduce the basis princple of GD, then we will introduce the mini-batch ADAM, which is the varients of GD that we will used in this thesis project.

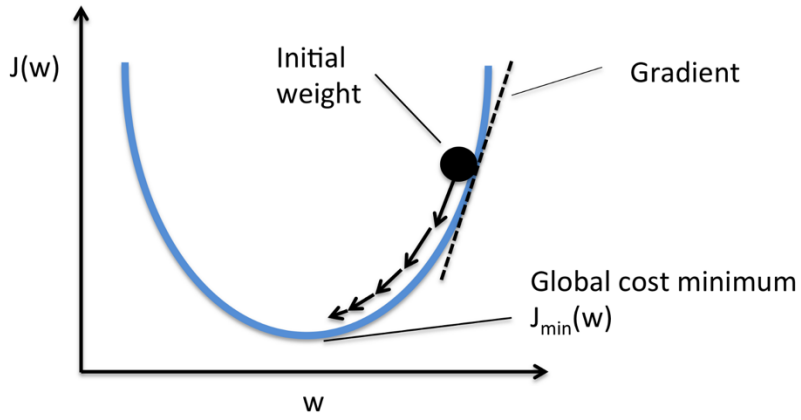

Fig. 7. Minimizing cost function J(w) by gradient descent

Fig. 7 shows the main idea of gradient descent for a single variable function $J(w)$. More commonly, for a function $E(x)$ with several variables, it is assumed that $E(x)$ is known and differentiable in the neighbourhood of a point θ. $E(x)$ declines fastest in the reverse direction of the gradient of $E(x)$ at θ, namely $-\nabla E(\theta)$. If the new value of θ is chosen as following

$$\theta_{new} = \theta_{old} - \gamma \nabla E(\theta_{old}) \tag{26}$$

for γ small enough, then $E(\theta_{new}) \leq E(\theta_{old})$. In this formula, the term $\gamma \nabla E(\theta_{old})$ is removed from the old value of θ because we need to go towards the inverse direction of the gradient to further minimize the cost function. For this reason, if we start with θ_0 randomly and consider the series $\theta_0, \theta_1, \theta_2, \ldots$ s.t.

$$\theta_{t+1} = \theta_t - \gamma \nabla E(\theta_t), t \geq 0 \tag{27}$$

Then

$$E(\theta_0) \geq E(\theta_1) \geq E(\theta_2) \geq \ldots \tag{28}$$

and ideally the series $\{\theta_t\}$ converges to a local minima. Here, γ is named the learning rate and the value of γ can be adjusted after per iteration.

After making some assumptions to the function E (i.e, E is convex and ∇E is Lipschitz stability) and careful choices of the learning rate γ, it is guaranteed that the function E can convergence to a local minima. This process is illustrated in Fig. 8.

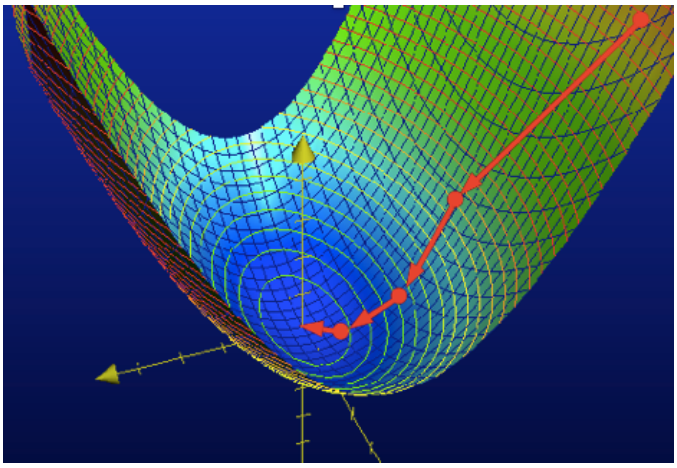

Fig. 8. Gradient descent process for multivariable function

In Fig. 8, the red arrow corresponds to the inverse direction of the gradient from a starting point. Besides, the direction of the red line is orthogonal to contour lines that pass the point. According to this figure, the final point is located at the bottom of the bowl, in which the function value is minimized.

Many variants of GD are widely used in the optimization of artificial networks, like Mini Batch GD (MBGD), Batch GD (BGD), Stochastic GD (SGD) and Adaptive Moment Estimation (ADAM). Among these methods, ADAM is a more adaptive technique. The main idea of ADAM is shown in equation (29). In which, γ is the learning rate, θ_t is the parameter we want to update, $\beta_1, \beta_2 \in [0,1]$ are exponential decay rates, $\nabla E(\theta_t)$ is the gradient of θ at time t, m_t, v_t are first and second momentum new defined. For more details, please consult the paper of Kingma and Ba[35].

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla E(\theta_t) \tag{29}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)\nabla E(\theta_t)^2$$

$$\theta_{t+1} = \theta_t - \frac{\gamma}{\sqrt{v_t + \varepsilon}} \frac{\sqrt{1-\beta_2^2}}{1-\beta_1^2} m_t$$

ADAM does not require too much memory and is effective in computation. For above reason, ADAM is currently widely used in complex neural networks like LSTM.

In our thesis project, we will use ADAM to update the parameters shown in Table 2. Suppose we have one year PM2.5 hourly concentration data (365*24=8760) and we use it to train our LSTM model. There are 3 possible ways to update parameters with ADAM.

The first method is called Stochastic Gradient Descent (SGD). We calculate the gradient and update the parameter for each sample in the training data. It is also called online machine learning algorithm because the system update for each example. The frequent update indeed has its benefits. However, updating the weights in high frequency is computationally expensive and consumes more time to train the model on larger collection of data. Besides, such an update may lead to a noise for gradient. Therefore, SGD are not suitable for large data set and LSTM.

The second method is called Batch Gradient Descent (BGD), which is an opposite approach of SGD. We still calculate the error for every sample in the training set. The average error of all training examples are used to calculate the gradient and fewer updates are done to the system. However, BGD may result in the early convergence to a less locally optimum because of the more stable error gradient.

The third method is called Mini-batch Gradient Descent (MBGD), which aims at balancing the robustness of SGD and the time-efficiency of BGD. We can split the training dataset into groups and these groups are called mini-batches. We only update parameters after the errors (sum or average of the gradient) of one mini-batch are calculated. MBGD updates frequently than BGD, which enables a much more robust convergence. Moreover, MBGD is much more efficient than SGD. MBGD is the most common GD method in AI algorithms. In our thesis project, we will use mini-batch type of ADAM to update our parameters.

2.3.4 LSTM network structure

We already know the structure within a single LSTM unit and algorithms (BPPT and mini-batch ADAM) that are used to update the network. Now we introduce how LSTM units are used in a LSTM network and how can we control the memory ability of LSTM. Recalling the dimension of vectors we mentioned in the introduction of LSTM unit, that are $x_t \in R^d$, $h_t \in R^h$, $c_t \in R^h$. The dimension of x_t is easy to understand that the input vector to the LSTM unit has dimension d. But why the dimension of c_t and h_t is h? Can we set the value of h or it should be default value based on the dimension of input vector? Actually, we can set h manually and this value is normally called the number of neurons of a LSTM unit. And in the real world programming, d always takes same value as h by using a matrix-vector multiplication to map the original data to h-dimensional.

Suppose we have a prediction task that predicting PM2.5 of next hour mainly based on the PM2.5 data of past 24 hours and we have one year PM2.5 data. Suppose we select 8 features including current PM2.5, wind direction, temperature that we think are important data to predict the PM2.5 of next hour. Then for this task, our input vector has 8 dimensions ($d = 8$), and output vector has 1 dimension (single PM2.5 value of next hour). And we want to build a prediction model based on LSTM network.

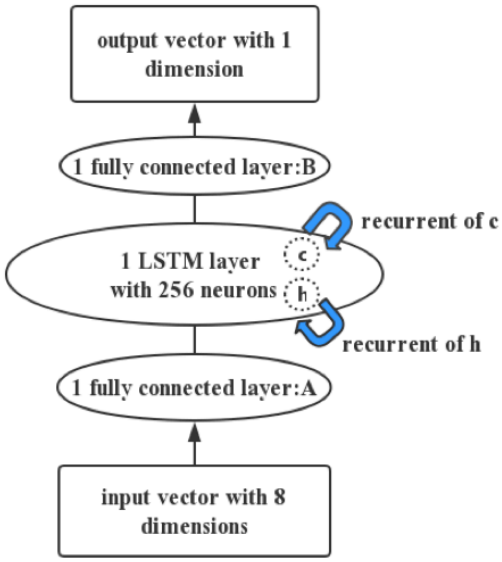

Fig. 9 (a)

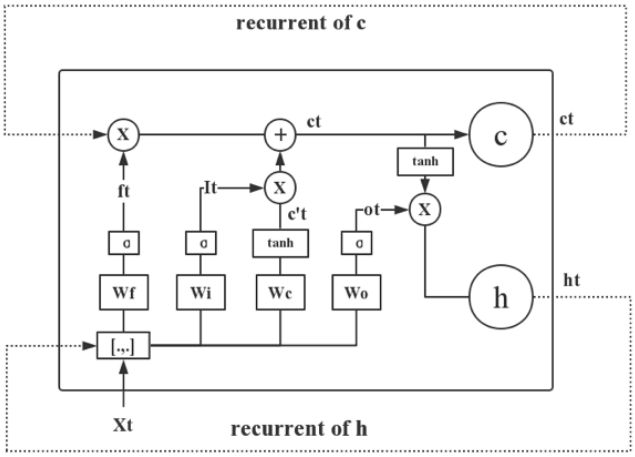

Fig. 9 (b)

Fig. 9 (a) is a sketch for LSTM network with 2 FC layers and 1 LSTM layer. (b) is the recurrent LSTM unit within a LSTM layer

A simple and common structure is shown as Fig. 9 (a), a LSTM network with 2 fully connected (FC) layers and 1 LSTM layer. As an example, here we simplify use 1 LSTM layer with 256 neurons ($h = 256$) based on the assumption that this size is reasonable for this task. The FC layer A are used for mapping the input data from 8 dimensions to 256 dimensions by matrix-vector multiplication. And the same for the FC layer B, mapping the output of LSTM (256 dimensional vector) to 1 dimension. Now we only left the explanation of the part of LSTM layer. We know that LSTM is a variant of RNN, that means LSTM also has recurrent structure. Fig. 9 (b) show the recurrent structure for a LSTM unit. For easy understanding, we unfold this recurrent LSTM unit, and the LSTM structure in Fig. 9 (a) can be shown as Fig. 10.

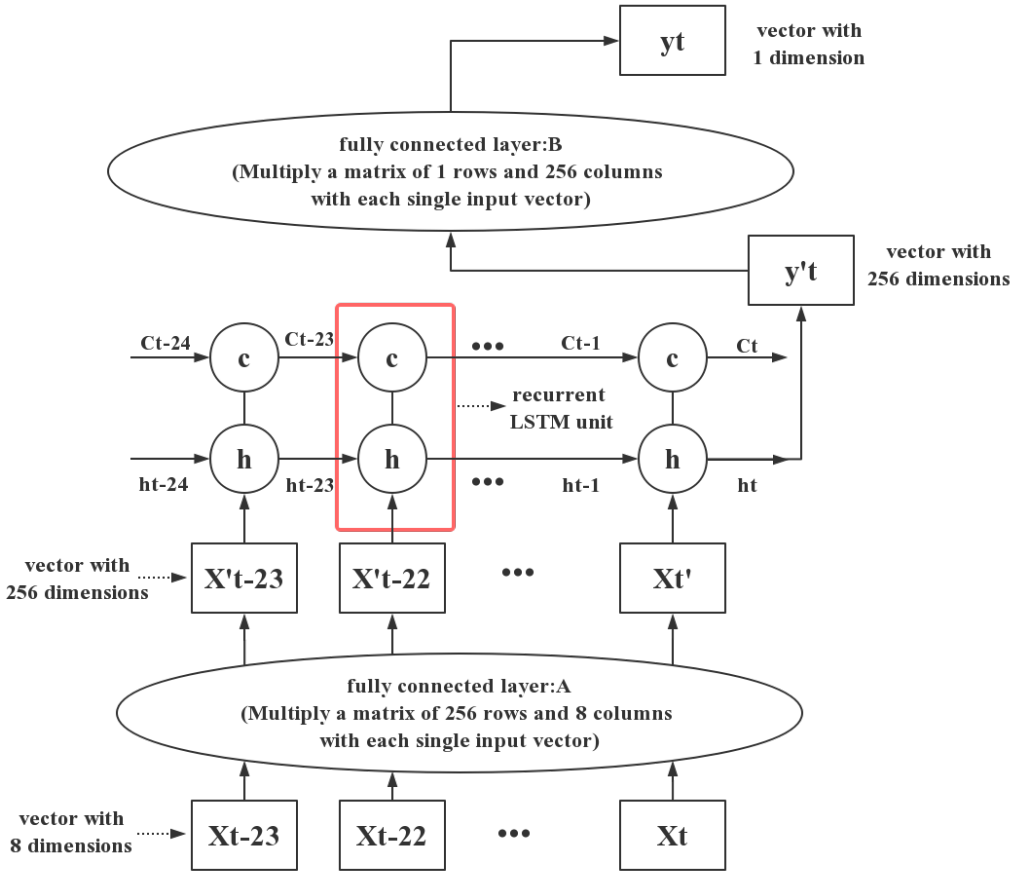

Fig. 10. Recurrent structure for a LSTM network with 2 FC layers and 1 LSTM layer

Before we explain the recurrent structure in Fig. 10, we should remember that the network only receives and propagates a single vector during each time. So in Fig. 10, the LSTM network first receives the 8-dimensional vector x_{t-23}, and maps it to 256-dimensional vector x'_{t-23} by FC layer A. Then x'_{t-23} is propagated to the LSTM unit. By the recurrent structure shown in Fig. 9 (b), $x'_{t-23}, c_{t-24}, h_{t-24}$ are received by the LSTM unit and c_{t-23}, h_{t-23} are calculated. Since this is the first calculation for LSTM unit, c_{t-24}, h_{t-24} are set as default values. Now we finish the calculation of the LSTM unit for one time step. Since we want the predict the PM2.5 of next hour mainly based on the PM2.5 data of past 24 hours, we don't give the prediction to next layer now and continue to receive the x_{t-22}. Same as before, $x'_{t-22}, c_{t-23}, h_{t-23}$ are received by the LSTM unit and c_{t-22}, h_{t-22} are calculated. Then we continue to receive x_{t-21}.We do this for 24 times and the memory about the past is continually propagated by the cell state. In the last time, x'_t, c_{t-1}, h_{t-1} are received by the LSTM unit and c_t, h_t are calculated. Since we already calculate 24 times and the predicted value h_t is calculated by the information of past 24 hours, now we send the predicted value y'_t (equal to h_t) to the next layer and give the prediction of

y_t through FC layer B. Now we finish the first prediction. Moreover, the value stored in the cell state *c* and hidden layer *h* are initialized to their default value to get prepared for next 24-time steps calculation.

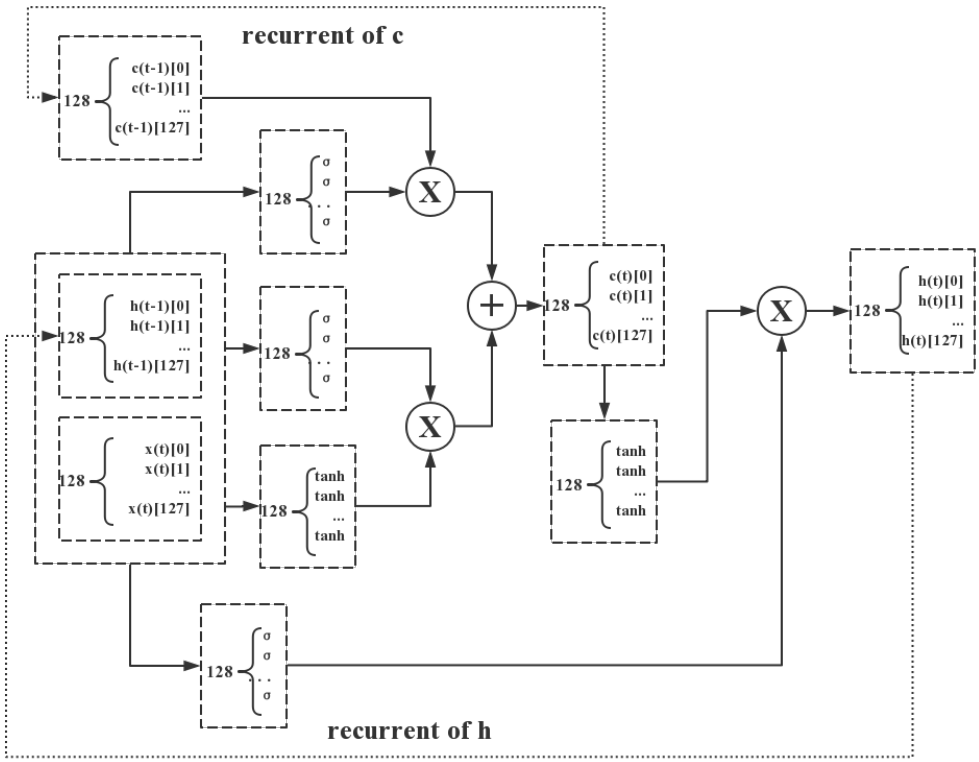

Fig. 11. Calculation process within a LSTM unit

Fig. 11 shows the calculation process within a LSTM unit and we could see the relation between the neurons and LSTM units. After the calculation of y_t, next step is the prediction on y_{t+1} based on the value of $x_{t-22}, x_{t-21}, ..., x_t, x_{t+1}$. Similarly, the LSTM network first receives the 8-dimensional vector x_{t-22}, and map it to 256-dimensional vector x'_{t-22} by FC layer A. Then x'_{t-22} is propagated to the LSTM unit. By the recurrent structure shown in Fig. 9 (b), $x'_{t-22}, c_{t-23}, h_{t-23}$ are received by the LSTM unit and c_{t-22}, h_{t-22} are calculated. Since this is the first calculation for LSTM unit, c_{t-23}, h_{t-23} are set as default values. Same as before, in the last time, x'_{t+1}, c_t, h_t are received by the LSTM unit and c_{t+1}, h_{t+1} are calculated. Since we already calculate 24 times and the predicted value h_{t+1} is calculated by the information of past 24 hours, now we send the predicted value y'_{t+1} (equal to h_{t+1}) to the next layer and give the prediction of y_{t+1} through FC layer B. Now we finish the second prediction.

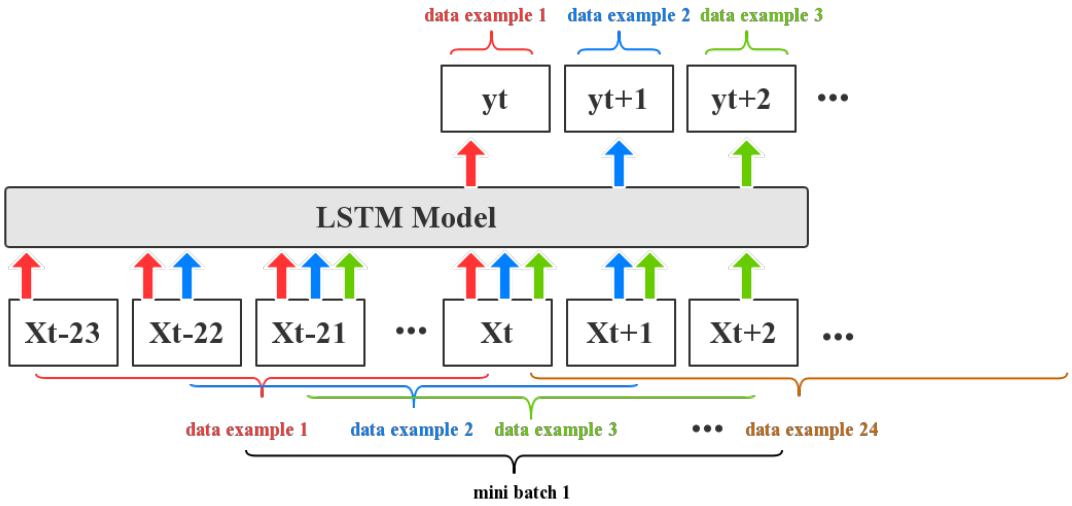

Fig. 12. Details for data example and mini batch (time step=24, mini batch size=24)

Since we set time step as 24 and we need the calculation based on 24 vectors, each group of 24 vectors is regarded as one data example. As we mentioned in chapter 2.3.3, we will use mini-batch ADAM which means we will put the data examples into groups and these groups are called mini-batches. We only update parameters after the errors of one mini-batch are calculated. Fig. 12 shows the details of data example and mini batch when time step is 24 and mini batch size is 24. Since the mini batch size is 24, each mini batch has 24 data examples. For example, mini batch 1 contains from data example 1 to data example 24. Similarly, mini batch 2 contains from data example 25 to data example 48. In next chapter, we will build the air quality prediction and early warning system of extreme air pollution model based on the LSTM network.

3. PM2.5 concentration prediction and early warning system of extreme air pollution based on the LSTM

In this chapter, we will build the PM2.5 concentration prediction model and early warning system of extreme air pollution based on the LSTM. As we mentioned in section 1.1, we define extreme air pollution as the air condition with a very high value of PM2.5. We will start from the data collection and processing, then we will build the LSTM model to forecast the PM2.5 concentrations in the following 24 hours. Since we also care about the extreme conditions, we will also build an early warning systems of extreme air pollution by predicting the grade of extreme air pollution in next 24 hours based on LSTM.

3.1 Data collection and processing

3.1.1 Data collection

Since the air pollution problem is extremely serious in the capital of China, Beijing, in our thesis project, we choose Beijing as our research city. Hourly air quality information observed by twelve observation stations in Beijing from January 18, 2013 to April 18, 2018 were downloaded (http://data.epmap.org/). Besides, we also need to collect meteorological data that might affect air pollutant concentrations. Half-Hourly weather condition data of Beijing from July 18, 2013 to Oct 31, 2016 were obtained from the Qingyue data center (http://data.epmap.org/). Constraint by the available meteorological data, our experiment values use hourly air quality and weather condition from July 18, 2013 to Oct 31, 2016. Although these data are not the most recent, the research process of model structure and optimisation method are still the same for data of other time periods.

3.1.2 Descriptive statistical analysis and preprocessing

Before we start to build the model, we need to check the data first and make sure they are suitable for data mining. We do the data cleaning first, such as deleting the invalid records and filling in the missing values by linear interpolation, and then we present the descriptive statistics of the data in Table 3 (Since we have 12 monitoring stations, here we show the description of 1 station for brevity).

Table 3. Description of the data (station 1001A)

	count	mean	std	min	25%	50%	75%	max
Wind direction	20481	185.54	117.10	10	90	160	310	360
Wind speed (m/s)	26542	2.91	2.21	0	1	2	4	20
Temperature(℃)	26540	14.16	11.62	-16	4	16	24	42
Dew point(℃)	26530	3.22	13.98	-40	-8	5	16	27
AQI	26384	109.27	89.53	0	44	85	144	500
SO2(ug/m^3)	25568	16.30	24.15	1	2	6	19	411
NO2(ug/m^3)	26258	52.14	35.24	0	25	45	73	224
CO(mg/m^3)	25538	1.31	1.18	0	0.6	1.0	1.6	13
O3(ug/m^3)	24963	58.78	58.51	1	9	45	86	358
PM10 (ug/m^3)	18391	115.44	97.46	1	46	94	153	1000
PM2.5 (ug/m3)	25566	80.18	81.08	1	22	56	110	857

Since there were too many missing values in the PM10 data, we remove PM10 values in the experiment. We normalize each variable such that it takes value between zero and one, see Fig. 13. Finally, our dataset contains 26542 records for each station and each record consists of 10 variables (wind direction, wind speed, temperature, dew point, AQI, SO2, NO2, CO, O3, PM2.5). Moreover, 70% of the dataset contributes to the training set while the rest 30% is used for validation.

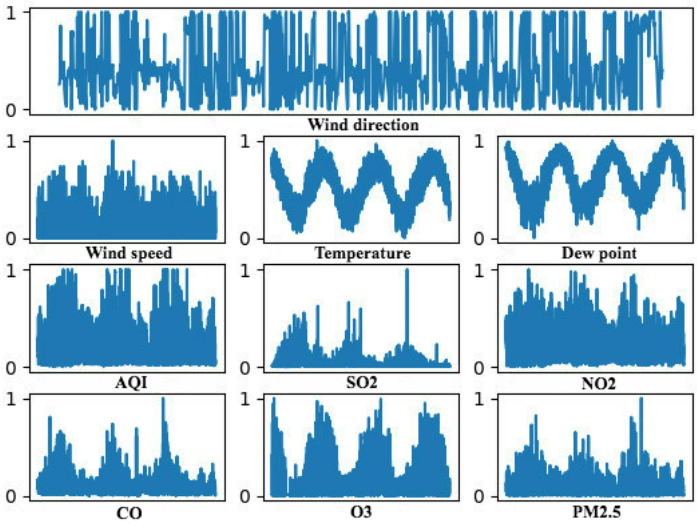

Fig. 13. Data after normalization. For each subfigure, X axis corresponds to the time line and Y axis represents normalized value

3.1.3 Pearson's and autocorrelation analysis

In section 3.1.2, we finished the basis preprocessing of data and we have 10 variables now, like the wind direction and wind speed shown in Fig. 13. In this section, we do Pearson's and autocorrelation analysis to measure the correlations among the data.

(1) Pearson's coefficients between PM2.5 and other variables

In this section, the Pearson's coefficient is used to measure the linear correlation among variable A and B. The result ranges from -1 to +1, in which +1 refers to absolutely positively linear dependence. 0 indicates that there is no linear dependence existed, and −1 represents perfectly negatively correlated.

Table 4. Pearson's coefficients between PM2.5 at timestamp *t* and other variables at timestamp *t*-1

R	PM 2.5	Wind direction	Wind speed	Temper -ature	Dew point	AQI	SO2	NO2	CO	O3
PM2.5	0.97	-0.05	-0.16	-0.15	0.03	0.93	0.51	0.68	0.79	-0.18

The Pearson's coefficients between different factors at timestamp *t*-1 and PM2.5 at timestamp *t* are shown in Table 4. Since the correlation value between the PM2.5 at timestamp *t* and the PM2.5 at timestamp *t*-1 is 0.97 (p-value < 0.05), it is demonstrated that there is a strong relationship between the current PM2.5 and the PM2.5 concentration of previous time. Besides, the Pearson's coefficients of AQI, SO2, NO2, CO are higher than 0.5 (p-value < 0.05), which indicates the PM2.5 data are more or less linear correlated with these factors. However, according to the Pearson's coefficients shown in Table 4, no obvious linear dependence can be founded between PM2.5 and the remaining features. This means that linear regression model is not suitable for this problem if we want to forecast PM2.5 according to these 10 variables. Based on the special gate-control recurrent structure and non-linear function, LSTM can deal with these non-linear time series prediction problems.

(2) Pearson's coefficients between stations

The spatial correlations of PM2.5 concentration among different observations are given in Table 5. According to Table 5, all the results are higher than 0.86 (p-value<0.05), which indicates there are strong spatial correlations among the selected observations. These correlations also lead to the optimization method used in section 3.2.2, in which the prediction is based on 12 stations' information including neighbouring correlated data.

Table 5. Pearson's coefficients between stations

R	No.1	No.2	No.3	No.4	No.5	No.6	No.7	No.8	No.9	No.10	No.11	No.12
No.1	1.00											
No.2	0.99	1.00										
No.3	0.97	0.97	1.00									
No.4	0.96	0.95	0.93	1.00								
No.5	0.95	0.96	0.96	0.95	1.00							
No.6	0.96	0.93	0.95	0.93	0.95	1.00						
No.7	0.91	0.88	0.95	0.94	0.96	0.95	1.00					
No.8	0.88	0.90	0.88	0.91	0.92	0.94	0.93	1.00				
No.9	0.90	0.91	0.91	0.89	0.93	0.95	0.94	0.93	1.00			
No.10	0.92	0.93	0.92	0.88	0.96	0.92	0.88	0.91	0.96	1.00		
No.11	0.94	0.92	0.87	0.91	0.91	0.87	0.86	0.90	0.93	0.90	1.00	
No.12	0.91	0.91	0.94	0.92	0.92	0.88	0.90	0.91	0.88	0.92	0.95	1.00

(3) Auto-correlation coefficients

In this section, the auto-correlation function is used to evaluate the temporal correlation among the air pollutants data. The auto-correlation value is defined as:

$$\rho_k = \frac{\mathrm{cov}\left(x(t), x(t+k)\right)}{\sigma_{x(t)}\sigma_{x(t+k)}} \tag{30}$$

in which $x(t)$ and $x(t+k)$ denote the PM2.5 values at corresponding time. $\sigma(\cdot)$ represents standard deviation and cov(·) refers to covariance. The auto-correlation values of PM2.5 concentration of station 1001A are shown in Fig.14.

With the increase of time-lag, there is a clear decrease in terms of coefficients, which reveals that the event of earlier time has less effect to present condition. Besides, the auto-correlation value is greater than 0.5 for the time-lag below 14. In the following works, these observations are helpful for choosing the suited time-lag.

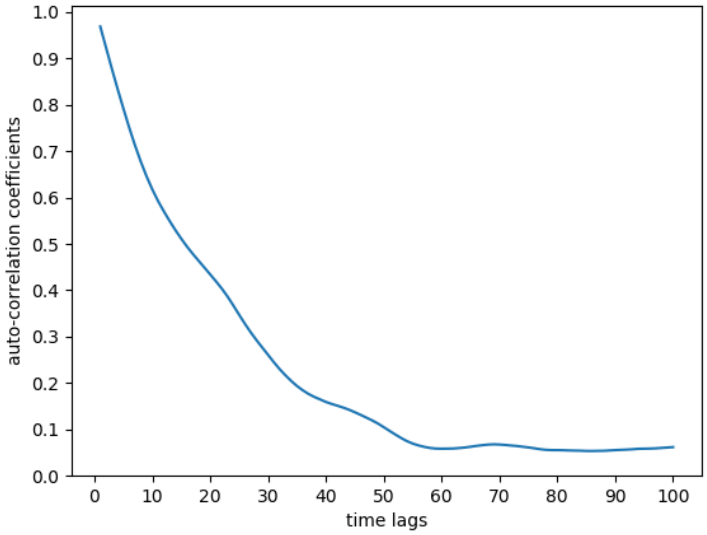

Fig. 14. The auto-correlation values of station 1001A

3.2 Real-time prediction model of PM2.5 concentration based on LSTM

In this section, we will build the real-time forecasting system of PM2.5 values based on LSTM. We first start from the 1-hour prediction task, that is predicting PM2.5 of next hour based on previous data. We will do parameter optimization to determine the best network architecture. In the training process, loss function is taken as Mean Square Error (MSE) to minimize expected loss. In the testing process, three indexes are employed to assess the prediction precision as shown in following equations.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - y_i^*\right)^2} \tag{31}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - y_i^*\right| \tag{32}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{\left|y_i - y_i^*\right|}{y_i^*} \tag{33}$$

In equation (31)(32)(33), y^* represents observed data and y_i represents model result and n is the size of test samples. Based on the optimal 1-hour prediction model, we will build 24-hour prediction model that is predicting PM2.5 concentration of next 24 hours.

3.2.1 1-hour prediction

(1) 1-hour prediction model

We first start from the 1-hour prediction problem, that is predicting PM2.5 of next hour based on previous data. We choose station 1001A as our research target and Fig. 15 shows the details of 1-hour prediction problem with time lag *r*.

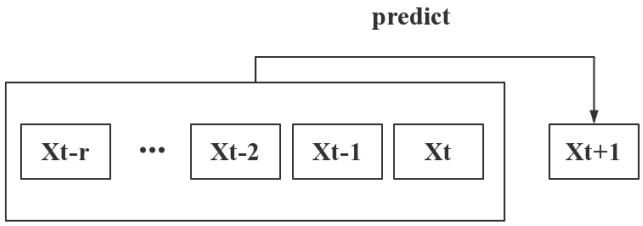

Fig. 15. Illustration of the 1-hour prediction problem with time lag *r*

From the LSTM model structure shown in Fig. 10, we know that several hyper parameters need to be determined first, like the time-lag, size of LSTM network and learning rates. In the following parts, we will investigate the influence of these parameters and search an optimal setting of parameters and the structure.

(2) Parameter optimization

In order to get optimal parameters, we need determine the model structure first. Based on existing research and experiences, neurons number of LSTM layers was selected from {8, 16, 32, 64, 128}. The number of LSTM layers was chosen from {1,2} that is, using 1 LSTM layer or 2 LSTM layers. Learning rate was selected from {0.01, 0.0005, 0.0001, 0.00005}. We use 2 FC layers in our research. We chose these settings according to the findings from some contrast tests, which shows these configurations are reasonable for our problem scale. And based on the autocorrelation analysis shown in Fig. 14, time-lag was chosen from {2, 6, 10, 14, 18, 22}.

We first researched the neurons number of LSTM layers and learning rate. The model structure was set as 1 LSTM layer and 2 FC layers and the time lag was set as 14 for experiment. The test result (RMSE of test data) is shown in Table 6.

Table 6. Effect of the number of neurons in LSTM layer and learning rate (RMSE of test data)

LSTM neurons / Learning rate	8	16	32	64	128
0.001	21.0105	22.2738	26.0959	23.9387	24.1670
0.0005	20.6172	21.9289	23.0614	27.8988	26.6133
0.0001	20.3021	21.1407	23.3050	26.3387	27.9382

0.00005	20.2744	**20.2264**	23.0846	26.4727	28.9341

From Table 6, we can see that the test result of LSTM model with 8 and 16 neurons were normally better than performance of model with larger number of neurons. As a comparison, we can check the PTB language model shown in the tutorial of Tensorflow (https://www.tensorflow.org/tutorials/recurrent), a famous LSTM based natural language processing model, which deals with 929,000 data records (each record is consists of 256 variables) by a LSTM model structure with 500-2000 neurons and could get satisfying results. However, our dataset contained 26542 records and each record consists of 10 variables, our data size is much more suitable for smaller LSTM model.

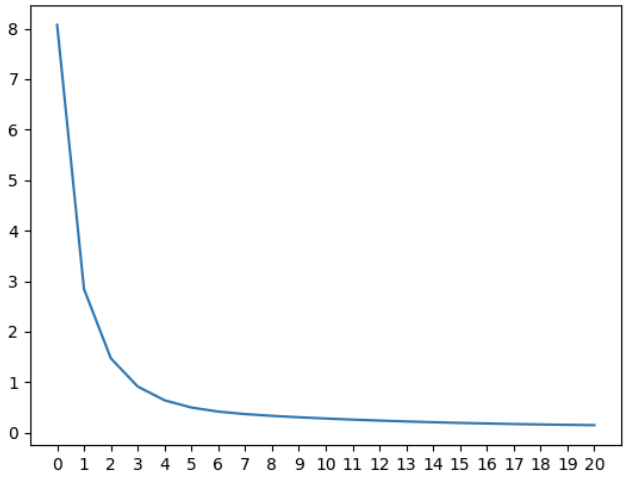

Fig. 16. Training loss during training process. Learning rate=0.00005, number of neurons=16. The vertical axis stands for the training loss and the horizontal axis stands for the number of training epoch.

For the learning rate, we could find that for LSTM model with larger size like model with 64 and 128 neurons, smaller learning rate performed more poorly than larger learning rate. This is possibly because model with more neurons has more parameters need to be optimised, and smaller learning rate could not guarantee a good learning speed and ability. Thus it's likely to trap into local optima which leads to a low convergence accuracy. For LSTM model with smaller size, we had opposite result. Larger learning rate speeds up the learning process and guarantees the learning speed, however, compared with the smaller learning rate, it's easy to skip extreme points and this might cause instability in the optimization process. The RMSE result also proved this point.

Based on the above analysis, in 1-hour prediction model, the number of LSTM neurons was set as 16 and learning rate was set as 0.00005. The training loss during training process is given in Fig. 16, the vertical axis stands for the training loss and the horizontal axis stands for the number of training epoch. An epoch is a measure of the number of times all of the training dataset are employed one time for parameter updates.

At the beginning of training, the loss was around 8 because all the parameters were set as initial values. However, after one training epoch the loss rapidly dropped below 3 and was stably kept below 1 after five epochs. Finally, we got a satisfying training loss of 0.1 after 20 training epochs. This proved that our LSTM model had strong learning ability and high efficiency in the prediction of PM2.5. To improve the performance, we continued to examine the influences of different time lags and Table 7 indicated the RMSE of test result.

Table 7. Effect of time-lag

Time lag	2	6	10	14	18	22
RMSE	28.2343	22.2157	**20.0768**	20.2264	21.2340	22.7687

The prediction performance shows that the LSTM model got sound performance when the time-lag was set as 10. Past research also found that smaller value of time-lag cannot ensure sufficient memory of existing trends. Thus, the model is unable to utilize the information adequately. Larger time-lag leads to the increasing of input information, which also add complexity of model and valuable patterns are hard to learn. For these reasons, we set time-lag as 10 which was a suitable choice regarding this problem.

Table 8. Effect of the number of LSTM layers

LSTM layers / Learning rate	1	2
0.001	22.0239	22.2349
0.0005	21.6289	21.1233
0.0001	21.1047	20.0121
0.00005	**20.0768**	**19.7863**

Finally, we focused on the research of LSTM layers. Time-lad was fixed to 10 and we set LSTM neurons as 16 based on the above results. There is a slight improvement for forecasting precision with the increasing of layers. As a result, the number of LSTM layers was set as 2 which can guarantee the forecast precision and efficiency. After determining an optimal

structure for the LSTM model, collected data were employed to train the system. The test data were used to evaluate the performance of the model and the results were given in Fig. 17.

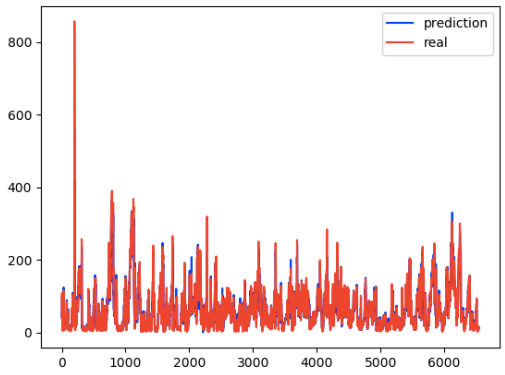

Fig. 17 (a) Predicted and real value of all the test data set (6542 records)

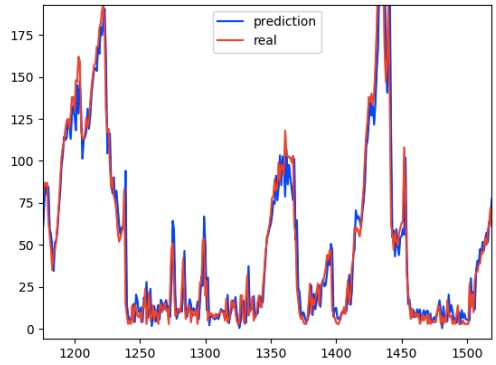

Fig. 17 (b) Predicted and real value of part of test data set (350 records)

Fig. 17. Predicted and real value of the test data set

Fig. 17 indicates that the forecasted value was universally close to the real condition and the RMSE of test data was 19.7863. Although this was a satisfying result, compared with the RMSE of Xiang Li's research (RMSE=12.60)[39], our RMSE was bigger than theirs. This was possibly because the training and testing dataset we use were different; thus the quality and feature of two dataset were also different. During the data processing in section 3.1.1, we also found that our data had thousands of missing values that need to be filled. This definitely affected the data quality. Besides, the models we built were different and the PM2.5 data they used were observed by 12 monitoring stations which provided more information to enhance the prediction precision. Although the method and data we use were different, compared with their result, our RMSE had the same order of magnitudes (19.79 vs 12.60). This proved that our model achieved satisfactory performance.

3.2.2 24-hour prediction

Now we focus on the 24-hour prediction problems, that is predicting PM2.5 of next 24 hours based on previous data. Obviously, historical information from various time period has various influence to the future of interest. As indicated in Fig. 18, the collected data within certain time lags *r* were used as input for 24-hour forecasting problems. Each solid line given in Fig. 18 represented one prediction problem. For the first 3 coming hours, we used different models for every hour. Continually, the next 4-24 h was split into three time lags (4-6, 7-12, and 13-24 h) and different models were trained separately to output the average value of each period.

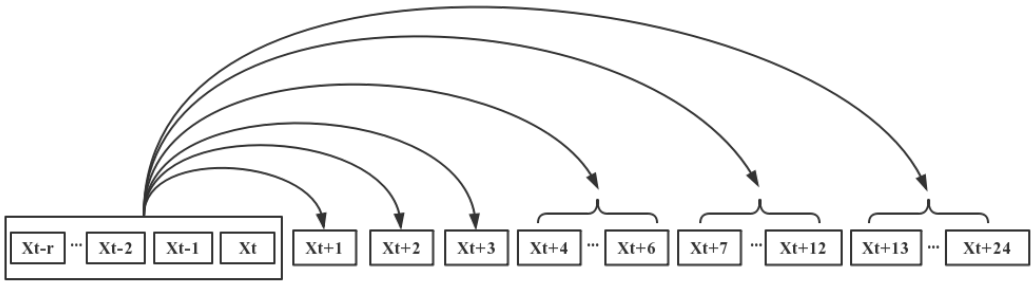

Fig. 18. Illustration of the 24-hour prediction problem with time lag *r*

In order to produce the best performance, the parameter for each task should be determined separately. However, aiming at simplifying the calculations, a fixed setting was applied in all tasks. There are 2 LSTM layers and 2 FC layers within this network. In this research, we just changed the value of time-lag. After utilizing grid search method, which was explained in section 3.2.1, the optimal structure for all problems were decided. The forecasting performance are given in Table 9.

Table 9. Structure and prediction accuracy of 24-hour PM2.5 concentration prediction

Task	Time lag	RMSE	MAE	MAPE(%)
1-h prediction	10	19.79	8.91	20.23
2-h prediction	10	21.14	10.13	23.89
3-h prediction	14	21.31	10.89	24.37
4 to 6-h prediction	15	26.33	14.15	32.60
7 to 12-h prediction	19	39.45	17.32	34.64
13 to 24-h prediction	28	49.41	21.93	40.20

According to Table 9, with the increase of time-lag, the best input time-lag increased as well while the forecast precision declined quickly based on the MAPE values, which changed from 20.23% to 40.20%. Since long-period forecasting problems are typically much more difficult, more information are needed compared with short-period problems. Our 24-hour prediction system showed sound precision including 13-24 h prediction task (RMSE=49.41).

3.3 Real time early warning system of extreme air pollution

3.3.1 Early warning system

In addition to ordinary air pollution, the phenomenon of extreme air pollution has also become more frequent in recent years (In section 1.1, we define extreme air pollution as the air condition with a very high value of PM2.5). Extreme air pollution poses serious influence to both environment and people survival. In this section, we will build an early warning systems of extreme conditions based on LSTM.

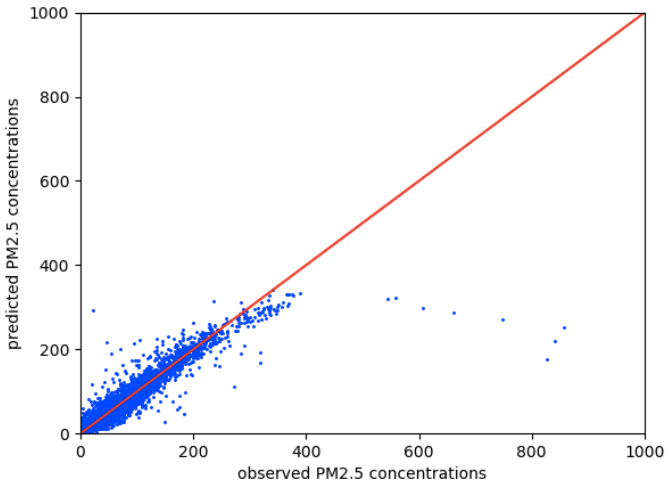

Fig. 19. Observed and predicted PM2.5 concentration for 1-h prediction model

From Fig. 19, we see that our PM2.5 prediction model performed well for most of data except the extreme air pollution with a high PM2.5 value. Although we got a satisfying RMSE (19.79), we cannot get accurate prediction for extreme conditions. Moreover, the predicted value of high PM2.5 concentration were always lower than the real value which cannot accurately monitor the extreme air pollution in advance. It is therefore necessary to build a model specially predicting the extreme air pollution and giving the early warning of extreme conditions.

Table 10 shows the standard rank of PM2.5 concentration in China, where rank I is the best air condition and rank VI is the worst. Since we define extreme air pollution as the air condition with a very high value of PM2.5, to simplify the problem, we here regard the highest PM2.5 value in the next 12 hours as the extreme air pollution of this period.

Table 10. Rank of PM2.5 concentration

Rank	Range of PM2.5 concentration
I	$0 \le x < 50$
II	$50 \le x < 100$
III	$100 \le x < 150$
IV	$150 \le x < 200$
V	$200 \le x < 300$
VI	$x \ge 300$

If we want to give an early warning of extreme air pollution to citizens 12 hours in advance, we can predict the rank of highest PM2.5 value of next 12 hours. By the predicted rank, we could know the most extreme level of PM2.5 pollution of next 12 hours and be prepared for it. In section 3.2, the LSTM model we built can predict the continuous PM2.5 concentration and that is a regression problem. Now we want to predict the discrete rank and this problem becomes a classification task. The loss function we use will be different and to make the output discrete we should also change the model structure.

3.3.2 Cross entropy and Softmax

In this section, we introduce the cross entropy and Softmax function that are always used in the classification model. And next section we will introduce how can we use these function to build the extreme air pollution classification model based on the LSTM.

(1) Cross entropy loss function

In the regression model like PM2.5 concentration prediction model we built in section 3.2, loss function is always set as MSE function. However, in the classification problem, cross entropy is normally used. For a given dataset x, the cross entropy for two distributions p and q is formulated as:

$$H(p,q) = -\sum_{x} p(x)\log q(x) \tag{34}$$

In information theory, the cross entropy is usually used to evaluate the amount of information required to distinguish an event from the dataset. We can understand that the cross entropy measures the distance between p and q, and the smaller cross entropy they have, the closer they are. We want to use the cross entropy to evaluate the accuracy of the LSTM classification algorithm, however, the output of the neural network is not necessarily a probability distribution.

To solve this problem, Softmax function is used to transfer the normal output of neural network to probability distribution.

(2) Softmax function

The softmax function generalizes the logistic function and is able to map a vector *y* of random real value to a vector $\sigma(y)$, in which every element ranges between 0 and 1 and the sum of all the elements is 1. The Softmax function is shown in equation (35)(36). In the next section, we will introduce how to use the Softmax function in our classification task.

$$\sigma : \mathbb{R}^N \to \left\{ \sigma \in \mathbb{R}^N \middle| \sigma_i > 0, \sum_{i=1}^{N} \sigma_i = 1 \right\} \tag{35}$$

$$\sigma(y)_j = \frac{e^{y_j}}{\sum_{n=1}^{N} e^{y_k}} \; for \; j = 1, \ldots, N \tag{36}$$

3.3.3 Model structure and result

(1) One-hot encoding

Recall that in the PM2.5 concentration prediction model, the input vector x_t has 10 dimensions which involves information like the wind direction and PM2.5 values at timestep *t*. The output vector y_t has 1 dimension which refers to the PM2.5 values of timestep *t*+1.

Table 11. One-hot encoding for the warning level and corresponding range of PM2.5

Warning level	One-hot vector	Range of PM2.5	Frequency	Percent
Level 1: Safe	(1,0,0,0)	$0 \le x < 50$	6692	25.22%
Level 2: Warning for sensitive groups	(0,1,0,0)	$50 \le x < 100$	7231	27.26%
Level 3: Warning for general populations	(0,0,1,0)	$100 \le x < 150$	5534	20.86%
Level 4: Serious warning	(0,0,0,1)	$x \ge 150$	7073	26.66%

In this extreme air pollution prediction model, the input data x_t is still the same as the PM2.5 concentration model. However, the output vector y_t represents the warning level for the PM2.5 condition of next 12 time lags (from time t+1 to time t+12) and it has 4 dimensions. In our

research, we divided the warning level into 4 parts which were encoded using one-hot encoding method shown in Table 11. We also counted the frequency and the percentage of each level in our dataset. We could see that the percentages of four warning level in our data are 25.22%, 27.26%, 20.86% and 26.66% respectively. These values are close to each other which means our data is suitable for 4-class classification problem.

(2) Model structure

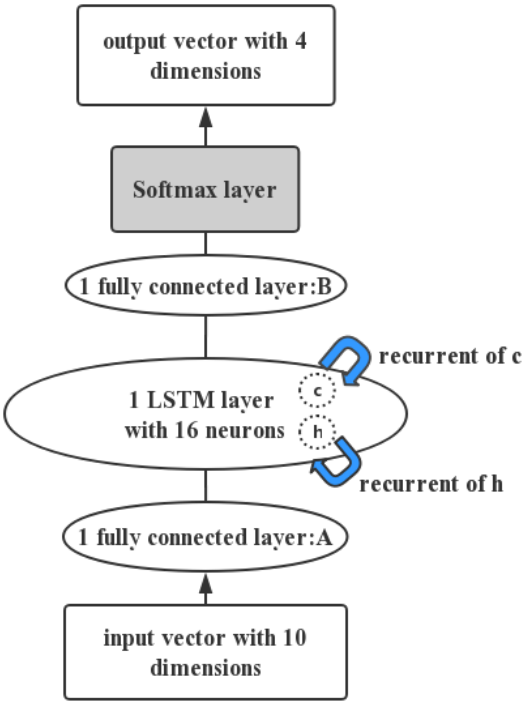

Fig. 20. Extreme air pollution prediction model based on LSTM

Since the real warning vector has 4 dimensions, the predicted warning vector should also have 4 dimensions. This can be achieved by changing the dimension of second fully connected layer from 1 neuron to 4 neurons. The normal output y_t^{normal} of the second FC layer is

$$y_t^{normal} = (y_t^1, y_t^2, y_t^3, y_t^4),\ where\ y_t^i \in R, i = 1,2,3,4\ . \tag{37}$$

However, to calculate the cross entropy between the predicted and the real warning vector, we need add a Softmax layer after the second FC layer to transfer the normal output to to probability distribution output y_t as shown in equation (38).

$$y_t = (y_t^1, y_t^2, y_t^3, y_t^4),\ where\ y_t^i \in [0,1], i = 1,2,3,4, and \sum_i y_t^i = 1 \tag{38}$$

Based on these operations, the model structure of extreme air pollution prediction is shown in Fig. 20. From Fig. 20 we could see compared with the PM2.5 prediction model, the extreme air pollution prediction model has one more Softmax layer. We know that the Softmax layer won't influence the data dimension and the dimension of output vector varies slightly, from 1 to 4. This supports that we could get satisfying result using similar LSTM structure as the optimal PM2.5 concentration prediction model that is 2 LSTM layers with 16 neurons and 2 FC layers, and set learning rate as 0.00005. Then we only change the time lag and find the optimal time lag for this model. Based on this target, the structure detail of extreme air pollution prediction model we used is shown in Fig. 21.

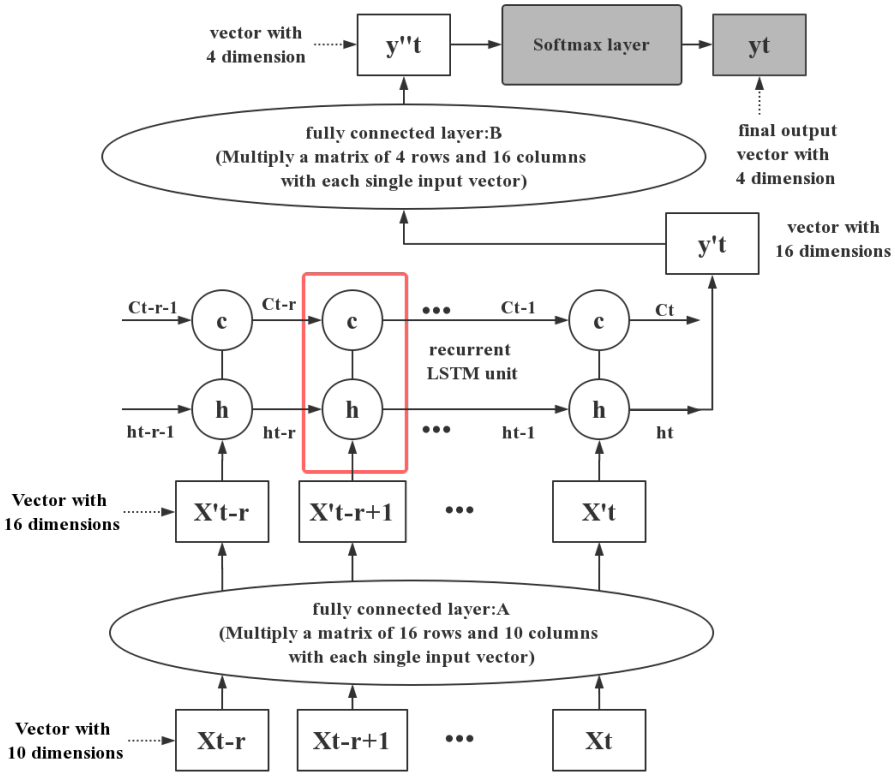

Fig. 21. Structure detail of extreme air pollution prediction model based on LSTM (number of LSTM neurons = 16, time lag = r)

(3) Receiver Operating Characteristic curve (ROC) and accuracy

Now we introduce how to assess the accuracy of our classification system. Receiver Operating Characteristic curve (ROC) and accuracy formula are often used to measure the forecast accuracy of a binary classification problem. We first introduce the basic principle and then explain how we apply these indicators in our research.

For a binary classification task, the outputs are labelled either as positive (p) or negative (n). Totally, there are 4 possible outputs. If the output of a forecast is p while the real condition is also p, then it is said to be a true positive (TP). If the real condition is n, then it is defined as a false positive (FP).On the contrary, a true negative (TN) appear if the output result and the real condition are n, and false negative (FN) is occurred if the model output is n while the real condition is p. As indicated in Table 12, these 4 outputs can be summarized in a confusion matrix.

Table 12. The confusion matrix of a binary classification problem

		True condition	
	Total data	Condition positive	Condition negative
Predicted condition	Predicted condition positive	TP	FP
	Predicted condition negative	FN	TN

Accuracy (ACC) of binary classification problem is defined as equation (39) which can show the model performance directly and higher accuracy indicates a better performance.

$$\mathrm{ACC} = \frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total data}} \tag{39}$$

Moreover, ROC curve is always used together to describe the classification model performance comprehensively. A ROC curve requires the true positive rate (TPR) and false positive rate (FPR) defined in equation (40)(41). The TPR represents the total amount of correct positive results appears among all positive examples. In addition, FPR represents the amount of incorrect positive outputs appear among all negative examples. By defining the *x*-axis and *y*-axis by FPR and TPR, a ROC space describes relative trade-offs between TP (benefits) and FP (costs).

$$\mathrm{TPR} = \frac{\sum \text{True positive}}{\sum \text{Condition positive}} \tag{40}$$

$$\mathrm{FPR} = \frac{\sum \text{False positive}}{\sum \text{Condition negative}} \tag{41}$$

Each confusion matrix corresponds to a point in the ROC figure. The point (0,1) of the ROC curve corresponds to the best forecasting performance. In addition, this point means 100% sensitivity (no FN) and 100% specificity (no FP). What's more, points above the diagonal indicate satisfactory classifications while points below this line show unsatisfactory classifications. After normalization of the unit, the Aera Under Curve (AUC) represents the chance that a classification model will classify a randomly selected positive sample greater than the negative sample we selected.

Table 13. The confusion matrix of 4-class classification problem

		True condition			
	Total data	Level 1	Level 2	Level 3	Level 4
Predicted condition	Level 1	True level 1			
	Level 2		True level 2		
	Level 3			True level 3	
	Level 4				True level 4

In our 4-class classification model, we can still use ACC and ROC curve to measure the model performance. For the ACC, it can be calculated by equation (42), in which True level are defined in Table 13.

$$\mathrm{ACC} = \frac{\sum \text{True level 1} + \sum \text{True level 2} + \sum \text{True level 3} + \sum \text{True level 4}}{\sum \text{Total data}} \tag{42}$$

As for the ROC curve and AUC for our 4-class classification model, we first derive the possibility matrix P and label matrix L of our test data shown in equation (43) and (44). In equation (43), P represents the prediction result of our model and p_{n1} represents the predicted possibility that data n belongs to warning level 1. Matrix L represents the true classification of our test data and l_n represents the one-hot vector of the warning level of data n.

$$P = \begin{pmatrix} p_{11} & \cdots & p_{14} \\ \vdots & \ddots & \vdots \\ p_{n1} & \cdots & p_{n4} \end{pmatrix} \tag{43}$$

$$L = \begin{pmatrix} one-hot\ vector\ l_1 \\ one-hot\ vector\ l_2 \\ \cdots \\ one-hot\ vector\ l_n \end{pmatrix} \tag{44}$$

For each warning level, we could derive 2 vectors from corresponding column in possibility matrix P and label matrix L. The first vector represents the predicted possibility of all test data belong to this level and the second vector represents the real possibility of all test data belong to this level. Based on these 2 vectors, we could derive the TPR and FPR of each level and draw corresponding ROC curve. Finally, we can take the average to get the final ROC curve of our 4-class classification model.

(4) Results and analysis

We used the model shown in Fig. 21 to train the data separately with different time lag until convergence. Then we used ACC and ROC curve to assess the classification accuracy. The ACC and AUC of our model with different time lag are given in Table 14.

Table 14. Effect of time lag

Time lag	ACC on test data	AUC
10	68.7%	0.722
15	75.3%	0.765
20	83.4%	0.801
25	**86.4%**	**0.837**
30	74.8%	0.749
35	63.2%	0.714
40	58.9%	0.705

According to the ACC and AUC (ACC=86.4%, AUC=0.837), our LSTM model can achieve the best performance when the time-lag r is equivalent to 25.

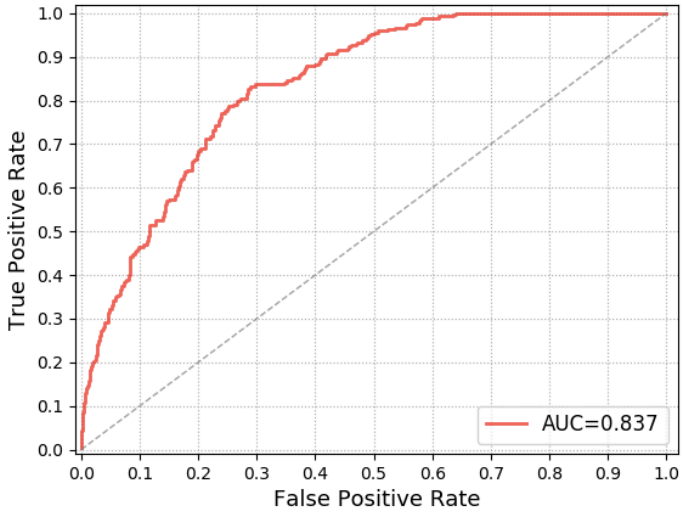

Fig. 22. ROC curve and AUC of the LSTM classification model with time lag r = 25

The ROC curve of optimal model is shown in Fig. 22. As we found in section 3.2.1, a small time-lag is not enough for our LSTM model and therefore the model cannot make adequately use of the LSTM to capture the required memory. This explains why the accuracy of model with small time lag is poor. However, very large input length leads to an increase of unrelated information, which makes the model much more difficult to extract valuable patterns.

Table 15. Confusion matrix of test result

		True condition			
	Total data	Level 1	Level 2	Level 3	Level 4
Predicted condition	Level 1	68	1	1	0
	Level 2	7	79	4	1
	Level 3	2	4	51	4
	Level 4	1	2	10	37

In our LSTM classification model, we find that when time lag exceeds 25, the model performance turns down. Finally, the setting time-lag was 25. This value was very suitable for our early warning model. The ROC curve also shows that the model can guarantee a good performance when the setting time-lag was 25. Besides, the confusion matrix shown in Table demonstrates that our classification model successfully gave the early warning of PM2.5 condition, even for the extreme conditions. Above findings proves that our model achieved satisfactory performance for the early warning of extreme air pollution.

4. Model Optimization

In this section, we will focus on model optimization techniques to improve the prediction performance. Since the model structure of PM2.5 prediction model and early warning model are similar, to simplify the research, we will only optimize the 1-h PM2.5 prediction model architecture we built in section 3.2.1. Four different methods will be done separately to test the influence and efficiency of optimization. Finally, we will combine these four methods together to derive an overall optimal model.

4.1 Parameter optimization: decreasing learning rate

In section 3.2.1, we found that high learning rate can speed up the convergence but it might be too large to skip extreme points. Low learning rate results in slow convergence but it can get better result. For this reason, if the learning rate was set at a relatively large rate in the beginning, so that the model can move forward to the extreme point quickly. Then the learning rate was decreased gradually to avoid skipping extreme points or causing instability in the optimization process.

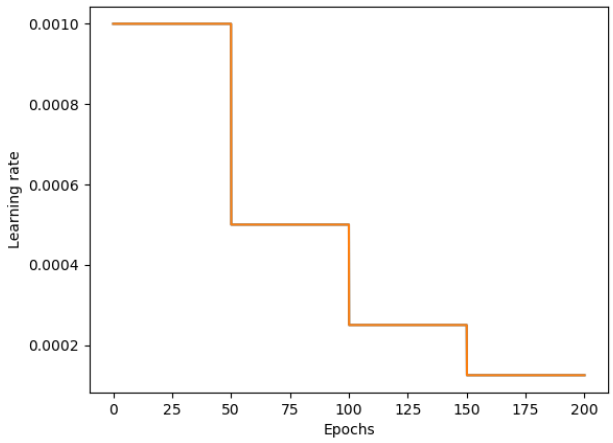

Fig. 23. Alternative schedule scheme of learning rate

An alternative schedule scheme of learning rate is given in Fig. 23. In the training process of LSTM, we reduced the learning rate by half by each 50 epochs until the convergence. As we mentioned in section 3.2.1, an epoch is a measure of the number of times all of the training dataset are used once to update the weights.

Table 16. Test result of decreasing learning rate

Learning rate	0.001	0.00005	Decreasing learning rate
RMSE	22.2349	19.7863	**18.6133**

Test result of adaptive learning rate is shown in Table 16. Compared with fixed learning rate, decreasing learning rate optimized both convergence speed and accuracy.

4.2 LSTM structure optimization: dropout regularization layer

In order to successfully apply NN, regularization is often required. Dropout, the most successful technique for regularizing NN, is often used in LSTM network to improve performance[36]. With the application of Dropout, input and recurrent connection to LSTM layers are probabilistically removed from the system during training. When keep probability is 1, the dropout layer has no effect on the network because every LSTM units can join the activation and weight updates. When keep probability is 0.5, each LSTM unit has a probability of 0.5 joining the activation and weight updates during training. A suitable dropout layer and keep probability can reduce overfitting and improve model performance.

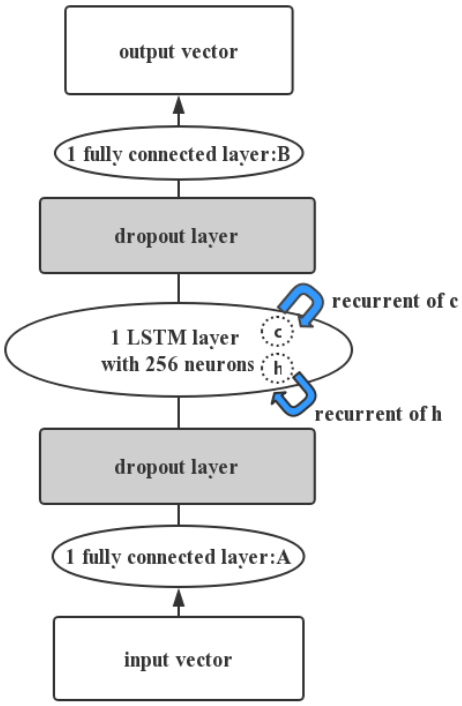

Fig. 24. LSTM model structure with dropout layer

To test the effect of dropout layer, we added it between the LSTM layer and the FC layer. The LSTM model structure with dropout layer is shown in Fig. 24. Then we tested different keep probability and results are given in Table 17.

Table 17. Test result of dropout layer with different keep probability

Keep probability	1.0	0.9	0.8	0.7	0.6	0.5
RMSE	19.7863	19.3212	**19.0177**	20.3890	22.2123	24.0922

When keep probability was set as 0.9 or 0.8, the test result was slightly better than the performance of LSTM model without dropout layer (RMSE 19.3212/19.0177 vs 19.7863). However, if we set keep probability as 0.7 or lower, the RMSE increased. This was probably because our model size was relative small. Too much dropout made our model lose important information and it became hard to learn the details.

4.3 Data optimization: use of seasonal information

Air pollution are known to change with season and the time of a day, and this information may be used to improve the prediction performance.

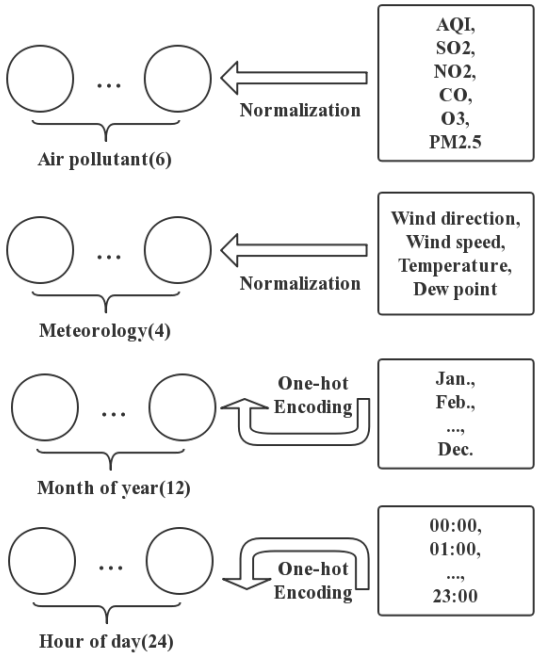

Fig. 25. Input data with seasonal information

In our 1-h PM2.5 concentration prediction model, our input data has 10 dimensions, which include the information about air pollutant and meteorology. We want to improve the model performance by using auxiliary data to provide more useful information. Since the concentration of PM2.5 varies with month and the time of a day, we increased the seasonal information in our data set. We used one-hot encoding and changed categories into binary code. For instance, there are 12 months, and with the usage of one-hot encoding, each monthly indicator can be converted into a vector of 12 dimensions (e.g., [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0] stands for the May). As a model improvement, this seasonal and time assistant information was concatenated into the output of the LSTM for learning.

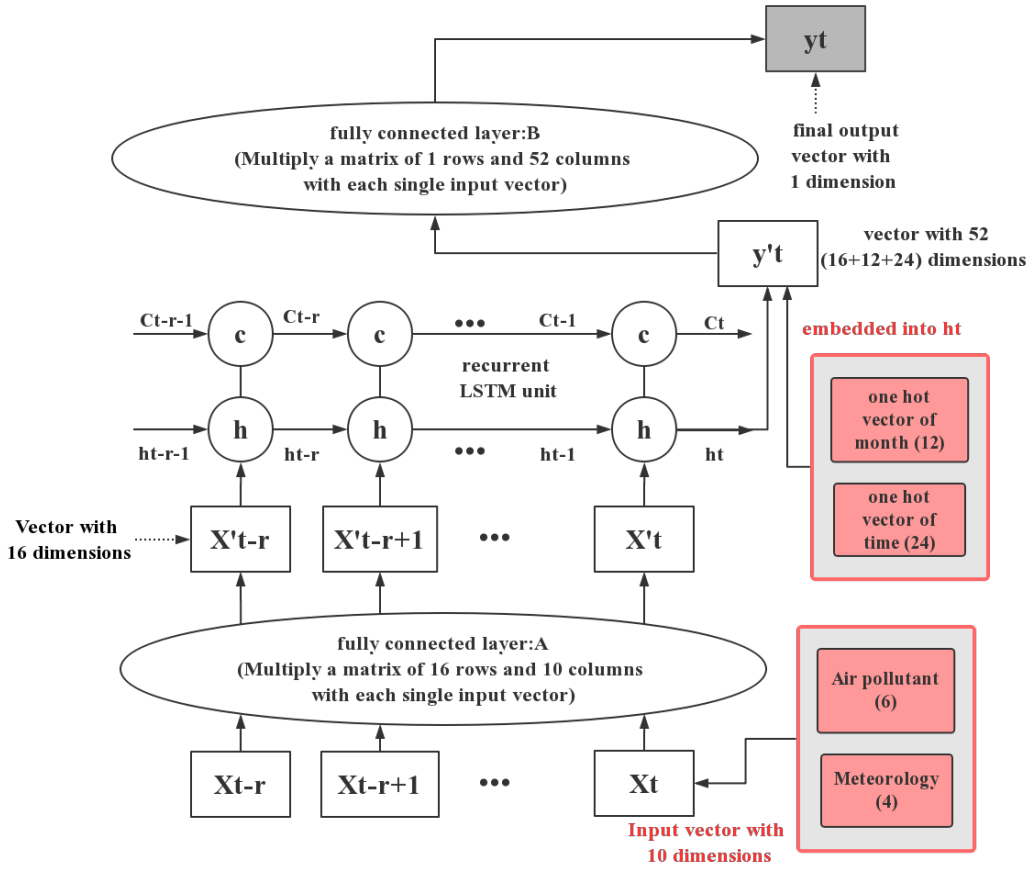

Fig. 26. 1-h PM2.5 prediction model with seasonal information

The 1-h PM2.5 prediction model structure with seasonal data is shown in Fig. 26. As shown in Fig. 26, the seasonal and time assistant information were concatenated into the output of the LSTM to add extra patterns. We did this mainly because high dimensional seasonal information already has obvious periodic trends and it's more efficient and easier for the normal fully connected layer to learn this feature. However, it's time consuming and expensive for LSTM to learn and store this obvious periodic trends from high dimensional seasonal information. That

is why we didn't put these data into the input variable of the LSTM unit. Since the input vector of the LSTM unit was still the same, only the dimension of the fully connected layer B increased. Using the same structure, we got a RMSE of 17.4531 which obviously improves the model performance compared with the model training on original data (RMSE of 19.7863). This suggests that seasonal information can be useful to improve the prediction accuracy of our LSTM model.

4.4 Data optimization: use of 12 monitoring stations' information

There are 12 air condition monitoring stations in Beijing, named from station 1001A to station 1012A. In our previous research, we focused on the PM2.5 prediction of station 1001A and only used the PM2.5 records of station 1001A. Now we want to improve the prediction accuracy for station 1001A by using information of all the 12 stations.

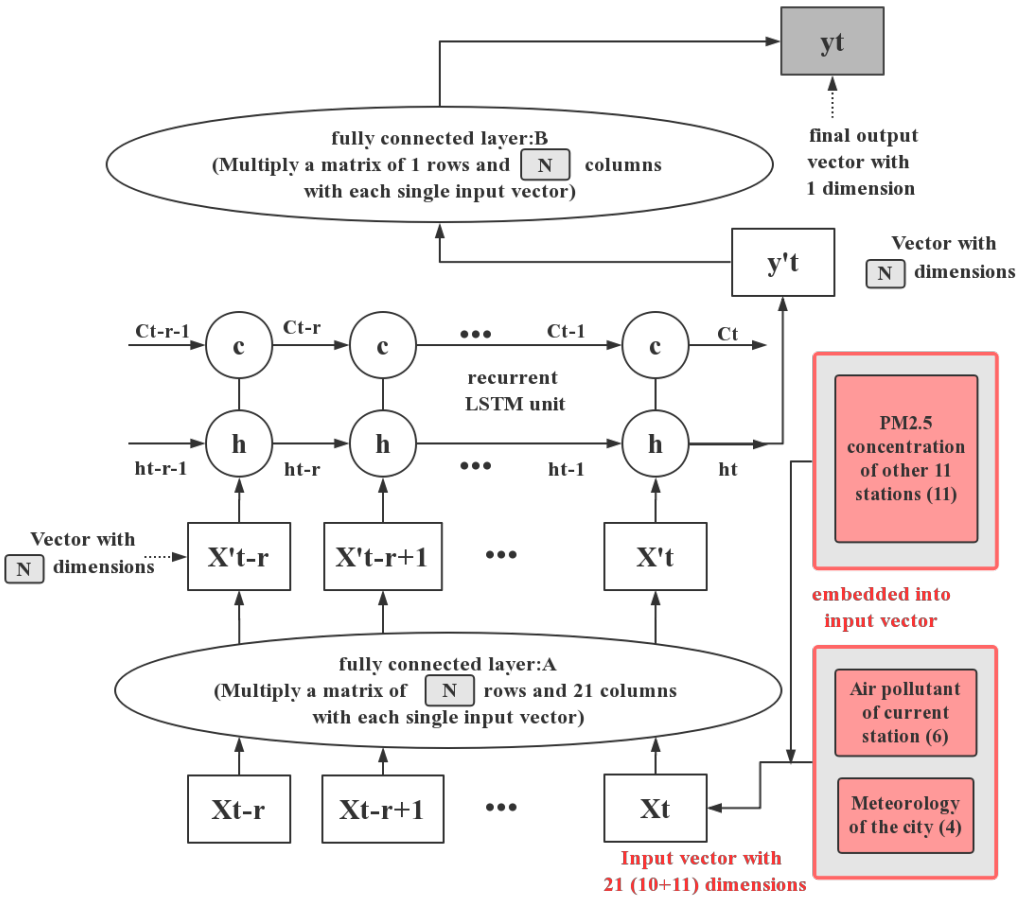

Fig. 27. 1-h PM2.5 prediction model with the information of 12 monitoring stations

Recall that in section 3.1.3, the spatial correlation of PM2.5 among the selected observations is given in Table 5. Since all coefficients are above 0.86 (p value<0.05), the PM2.5 data show

strongly correlations among the stations. This high spatial relation leads to the optimization method we will use in this section, predicting one station's PM2.5 based on 12 stations' information because nearby data can optimize the model.

Fig. 27 shows the structure of 1-h PM2.5 prediction model with the information of 12 monitoring stations. Recall that in the optimized model with seasonal information, we concatenated the seasonal and time assistant information into the output of the LSTM to add extra features. We did that mainly because high dimensional seasonal information already has obvious periodic trends and it is more efficient and easier for the normal fully connected layer to learn this feature.

In this time, the assistant information is about the PM2.5 at the rest observations. There is no obvious trend in the PM2.5 data and the normal fully connected layer cannot exploit and store the memory of the time series. For this reason, we used LSTM unit to learn and store this complex time series feature and the PM2.5 data of other 11 monitoring stations were embedded into the input data of the LSTM unit. Since the dimension of input vector increases from 10 to 21, we test different number of LSTM neurons based on the optimal 1-h PM2.5 prediction model in section 3.2.1 and the results are given in Table 18.

Table 18. Test result of model with different number of neurons

Number of neurons	16	28	40	52	64
RMSE	24.7831	19.0327	**16.4325**	23.3341	26.2343

Using the LSTM structure shown in Fig. 27 with 40 neurons, we got a RMSE of 16.4325 which effectively improved the model performance compared with the model training on original data (RMSE of 19.7863). This proves that PM2.5 concentration of nearby stations can effectively improve the prediction accuracy of our LSTM model. Besides, compared with optimal 1-h PM2.5 concentration prediction model in section 3.2.1, the optimal number of LSTM neurons increases from 16 to 40, this is possibly because the dimension of our input vector increases from 10 to 21 dimensions as well and model need more LSTM neurons to learn the feature among the data.

4.5 Comparison and combination of optimization methods

From section 4.1 to section 4.4, we done four different optimization methods separately based on the original 1-h PM2.5 prediction model and the RMSE on test data decreased accordingly. Intuitively, we wonder what if we combine all these optimization methods together to optimize the 1-h PM2.5 prediction model. We combined these four optimization together, that is using decreasing learning rate based on the LSTM structure with dropout layer shown in Fig. 24. Similarly, the information about 12 stations were embedded into the input data of the LSTM unit and the seasonal and time auxiliary data were concatenated into the LSTM output feature vector. Using the same experimental procedures as before, we got a much lower RMSE of 14.1937 when the number of LSTM neurons was set as 40. The comparison of different optimization method is shown Table 19.

Table 19. Comparison of different optimization method

Optimization method	RMSE on test data
Original 1-h PM2.5 prediction model	19.7863
Parameter optimization: decreasing learning rate	18.6133
LSTM structure optimization: dropout regularization layer	19.0177
Data optimization: use of seasonal information	17.4531
Data optimization: use of 12 stations' information	16.4325
Combination optimization	14.1937

According to Table 19, there are three main conclusions. Firstly, compared with original 1-h PM2.5 prediction model, all the optimization method exhibited better prediction performance. This finding proved that all these optimization techniques were suitable for the LSTM model and can improve the prediction performance accordingly. Second, compared with the parameter and LSTM structure optimization, data optimization methods produced better forecasting from RMSE perspectives. This result suggested that the optimization on the data can more efficiently help the LSTM model to capture spatiotemporal correlations and to get better performance. Table 19 also indicates that the combination optimization performed better than any single optimization method, which suggested that we can use some effective optimization methods together to increase the prediction precision of LSTM model.

5. Comparison of experiments

Random forests (RF) as an ensemble algorithm can be applied for both classification and regression. The key of RF is to construct several decision trees during training. For classification, the output is the mode of the classes of the individual trees while the output for regression is mean prediction of the decision trees. Tin Kam Ho used the random subspace method to build the first algorithm for RF. This algorithm made a combination between bagging and random feature selection [13].

RF is a powerful learning algorithm with high accuracy and can be employed efficiently on large datasets. RF has many advantages. Firstly, it can deal with thousands of features without variable selection. Secondly, because of the building progress, it can produce an unbiased estimate of the generalization error internally. Besides, RF can estimate missing data efficiently and keep accuracy when many data missed. In order to evaluate the performance of the LSTM model, we compare its performance of 24 hour PM2.5 prediction with the RF model in the beginning.

Table 20. Comparison of the PM2.5 prediction performance of the LSTM and RF

Task	RMSE of LSTM	RMSE of RF
1-h PM2.5 prediction	19.79	19.11
2-h PM2.5 prediction	21.14	23.45
3-h PM2.5 prediction	21.31	26.57
4 to 6-h PM2.5 prediction	26.33	32.60
7 to 12-h PM2.5 prediction	39.45	44.64
13 to 24-h PM2.5 prediction	49.41	56.20

The training and test datasets used in the LSTM prediction were applied for the RF model during training and testing. We trained the RF model mainly based on the RF regression package of python (RandomForestRegressor). The forecast precision of these two methods is given in Table 20 and we can get two useful findings. First, both LSTM and RF can get satisfying prediction performance in the PM2.5 prediction tasks, especially for 1 hour PM2.5 prediction. Secondly, we found that the LSTM model exhibited better prediction performance for long time prediction tasks from RMSE perspectives. This observation agrees with pass research (Li et al., 2016), where the LSTM model was demonstrated very appropriate for modelling complicated

spatiotemporal relations. Table 20 proves that our LSTM model can capture spatiotemporal correlations much more efficiently for long term time series prediction tasks.

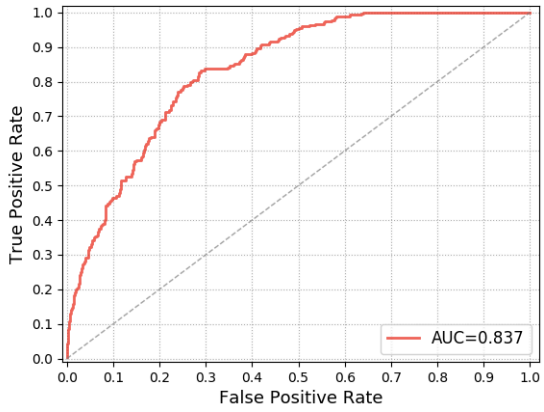

Fig. 28 (a) ROC curve of LSTM

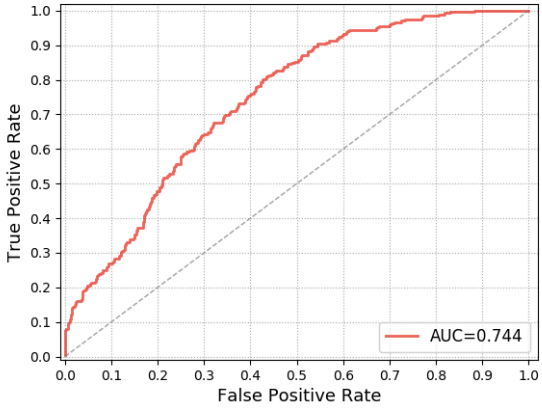

Fig. 28 (b) ROC curve of RF

Fig. 28. Comparison of the ROC curve of early warning system of extreme conditions.

Additionally, in order to assess the precision of our LSTM model on the prediction of extreme conditions, we also compare our early warning system with the RF model. We trained the RF model mainly based on the RF classification package of python (RandomForestClassifier). The comparison of ROC curve is given in Fig. 28 and Table 21 gives the comparison of confusion matrix.

Table 21. Comparison of confusion matrix of LSTM and RF (result of LSTM is in red bold font style)

		True condition			
	Total data	Warning level 1	Warning level 2	Warning level 3	Warning level 4
Predicted condition	Warning level 1	**68**|64	**1**|4	**1**|3	**0**|1
	Warning level 2	**7**|6	**79**|73	**4**|10	**1**|3
	Warning level 3	**2**|6	**4**|5	**51**|49	**4**|6
	Warning level 4	**1**|2	**2**|4	**10**|4	**37**|32

From Fig. 28, we could see that the AUC of our early warning system of extreme air pollution is 0.837, which is significantly higher than the AUC of RF (AUC of RF=0.744). And Table 21 shows that our early warning system based on LSTM has higher accuracy compared with the RF(ACC of LSTM=86.4%, ACC of RF=80.1%). These comparisons suggest that both the PM2.5 prediction model and the early warning system performed better than the traditional RF model. Besides, according to the structure character of the LSTM model, we can flexibly design

the model structure and the dimension of the output by change the number of neurons of network easily. Moreover, as we discussed in chapter 4, there are several optimization methods we can use to improve the prediction performance; thus our model not only has wide applicability but also performs well in multiple tasks.

6. Conclusions and recommendations

6.1 Conclusions

This thesis project developed an alternative PM2.5 concentration prediction model and early warning system of extreme air pollution based on the LSTM model and achieved satisfying performance. Since the air pollution problem is extremely serious in the capital of China, Beijing, we chose Beijing as our research city and hourly PM2.5 concentration and meteorological data were collected.

To research more deeply, we divided the task into two parts. The first task was predicting the PM2.5 concentration of next 24 hours and another one was building early warning system of extreme air pollution of next 12 hours. The LSTM model can model long period sequences and one could find best input time-lag easily. Although the first task was a regression problem and the second one was a classification task, LSTM can be applied for two tasks with similar architecture due to the flexible model structure of neural network.

To solve the first task, we started from the 1-hour prediction task, that was predicting PM2.5 of next hour based on previous data. We did parameter optimization to determine the best network architecture. In the training process, loss function was taken as Mean Square Error (MSE) to minimize the expected loss. In the testing process, in order to assess the performance of the model, three index were measured. Finally, we determined the optimal model structure which was 2 LSTM layers and 2 FC layers. The time-lag was set as 10 and the learning rate was 0.00005. The RMSE of test data was 19.7863 which was a sound precision.

Based on the optimal 1-hour prediction model, we then successfully built 24-hour prediction model that was predicting PM2.5 concentration of next 24 hours. Since previous information have various influence to the future of interest, the prediction tasks is split into several parts and we trained the model for each task. In order to produce the best performance, we decided the appropriate input time-lag for every problem. We found that with the increasing of time-lag, the best input time lag increased as well while the forecast precision declined quickly (MAPE ranged from 20.23% to 40.20%). Since long-period forecasting problems are typically much more difficult, more information are needed compared to short-period forecasting problems.

Our 24-hour prediction method produced sound performance, including the 13-24 h prediction task (RMSE=49.41).

Although we got a satisfying RMSE for the PM2.5 prediction task, we didn't get accurate prediction for extreme conditions and that's why we continued to focus on the second task. Since we defined extreme air pollution as the air condition with a very high value of PM2.5, to simplify the problem, we regarded the highest PM2.5 value among 12 hours as the extreme air pollution of this period. If we want to give an early warning of extreme air pollution to citizens 12 hours in advance, we can predict the rank of highest PM2.5 value of next 12 hours.

For this reason, we divided the warning level into 4 parts which are represented by one-hot vector and the problem becomes a 4-class classification problem. Based on the optimal 1-hour PM2.5 prediction model, we changed the dimension of second FC layer and added a Softmax layer to transfer the normal output to to probability distribution. The loss function was taken as cross entropy between model outputs and the real warning vector. According to the ACC and AUC (ACC=86.4%, AUC=0.837), our LSTM model can achieve the best performance when the time-lag r is equivalent to 25.

To improve the prediction performance, we also focused on several model optimization techniques for the LSTM model and effectively improved the accuracy. Since the model structure of PM2.5 prediction model and early warning model are similar, to simply the research, each optimization method was done separately based on the optimal 1-h prediction model. We first focused on the parameter optimization and used an alternative schedule scheme of learning rate which optimized both convergence speed and accuracy (RMSE decreases from 19.79 to 18.61).

Then we focused on the LSTM structure optimization. Dropout, the most successful technique to regularize NN, was used in our optimized LSTM network to improve performance. We added the dropout layer between the LSTM layer and the FC layer and we tested different keep probability. We found that when keep probability was set as 0.9 or 0.8, the test result was slightly better than the performance of LSTM model without dropout layer (RMSE 19.3212/19.0177 vs 19.7863). However, if we set keep probability as 0.7 or lower, the RMSE increased. This probably because our model size was relative small and too much dropout made model lose important information and was hard to learn the details.

Finally, we focused on the data optimization. Seasonal information and PM2.5 data of nearby obervations were added to the dataset separately to test the influence of optimization. Using the

seasonal information, we got a RMSE of 17.4531 which effectively improves the model performance compared with the model training on original data (RMSE of 19.7863). This proved that seasonal information effectively improves the prediction accuracy of our LSTM model. Moreover, with the PM2.5 concentration of nearby monitoring stations, we got a much lower RMSE of 16.4325 which showed that the information of nearby PM2.5 concentration was much more important for improving prediction accuracy. Besides, we also combined these four optimization methods together and we got the lowest RMSE of 14.1937 when the number of LSTM neurons was set as 40. The combination optimization performs better than any single optimization method, which suggests that we can use some effective optimization methods together to improve the prediction precision of LSTM model.

To assess the performance of our LSTM model, we compared our PM2.5 prediction model with the Random Forest (RF) model first. Both LSTM and RF got satisfying prediction performance in the PM2.5 prediction tasks, especially for 1 hour PM2.5 prediction. However, the LSTM based model produced better prediction performance for long time prediction tasks according to the RMSE. These findings suggested our LSTM model can more efficiently learn spatiotemporal relations for long term time series prediction tasks.

In addition, to assess the accuracy of our LSTM model on the prediction of extreme conditions, we also compare our early warning system of extreme air pollution with the RF model. The comparisons show that our model performed better than the RF(AUC: LSTM=0.837, RF=0.744; ACC: LSTM=86.4%, RF=80.1%). These comparisons suggest that both the PM2.5 prediction model and the early warning system performed better than the traditional RF model. Besides, according to the structure character of the LSTM model, we can flexibly design the model structure and the dimension of the output by change the number of neurons of network easily; thus our model not only had wide applicability but also performed well in multiple tasks.

6.2 Possible improvements

Although we already got a satisfying prediction performance and found some optimization techniques, there are still some possible improvements.

1) From the perspective of data, we can also collect the data from more areas in the future research. If we want to predict the air quality of Beijing, we can also take the nearby cities' air quality data into account to improve the model performance.

2) Our data has some missing values and we use linear interpolation to fill in the missing values. In addition to simple linear interpolation, other method like polynomial interpolation can also be used to improve the approximation precision.

3) From the perspective of model architecture, there are also some variants of LSTM, like the Factorized LSTM (F-LSTM) which replaces matrix W by the product of two smaller matrices. The key assumption here is that W can be well approximated by the matrix of rank r. Such approximation contains less LSTM parameters than original model, therefore, can be computed faster and synchronized faster in the case of distributed training. This method can be considered when we have much more data to process and we want to improve the computing efficiency.

Appendix I. Bibliography

[1] Al-Helou, B. (2012). Air Pollutant Effects on the Environment of the Al-Hashimyeh Town. International Journal of Environmental Science and Development, pp.240-245.

[2] Liu, Y. Application of Deep Convolutional Neural Networks for Detecting Extreme Weather in Climate Datasets. (2016). eprint arXiv:1605.01156.

[3] McGovern, A. (2017). Using Artificial Intelligence to Improve Real-Time Decision-Making for High-Impact Weather. Bulletin of the American Meteorological Society, 98(10), pp.2073-2090.

[4] Qin, S. (2015). Seroprevalence and risk factors of Chlamydia abortus infection in free-ranging white yaks in China. BMC Veterinary Research, 11(1), p.8.

[5] Qin, S. (2014). Analysis and forecasting of the particulate matter (PM) concentration levels over four major cities of China using hybrid models. Atmospheric Environment, 98, pp.665-675.

[6] Sun, W. (2013). Prediction of 24-hour-average PM2.5 concentrations using a hidden Markov model with different emission distributions in Northern California. Science of The Total Environment, 443, pp.93-103.

[7] Cobourn, W. (2010). An enhanced PM2.5 air quality forecast model based on nonlinear regression and back-trajectory concentrations. Atmospheric Environment, 44(25), pp.3015-3023.

[8] Krizhevsky, A. (2017). ImageNet classification with deep convolutional neural networks. Communications of the ACM, 60(6), pp.84-90.

[9] Shin, D. (2017). Deep Learning Model for Prediction Rate Improvement of Stock Price Using RNN and LSTM. The Journal of Korean Institute of Information Technology, 15(10), pp.9-16.

[10] Ingenhoven, P. (2018). Comparison of Statistical and Deterministic Smoothing Methods to Reduce the Uncertainty of Performance Loss Rate Estimates. IEEE Journal of Photovoltaics, 8(1), pp.224-232.

[11] Feng, X. (2015). Artificial neural networks forecasting of PM2.5 pollution using air mass trajectory based geographic model and wavelet transformation. Atmospheric Environment, 107, pp.118-128.

[12] Chen, J. (2014). Seasonal modeling of PM2.5 in California's San Joaquin Valley. Atmospheric Environment, 92, pp.182-190.

[13] Ho, Tin Kam (1995). Random Decision Forests. Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282.

[14] Manders, A. (2017). Curriculum Vitae of the LOTOS-EUROS (v2.0) chemistry transport model. Geoscientific Model Development Discussions, pp.1-53.

[15] Stern, R.(2008). A model inter-comparison study focussing on episodes with elevated PM10 concentrations. Atmospheric Environment, 42(19), pp.4567-4588.

[16] Konovalov, I. (2009). Combining deterministic and statistical approaches for PM10 forecasting in Europe. Atmospheric Environment, 43(40), pp.6425-6434.

[17] Zhang, H. (2012). Prediction of ozone levels using a Hidden Markov Model (HMM) with Gamma distribution. Atmospheric Environment, 62, pp.64-73.

[18] García Nieto, P. (2013). A SVM-based regression model to study the air quality at local scale in Oviedo urban area (Northern Spain): A case study. Applied Mathematics and Computation, 219(17), pp.8923-8937.

[19] Ramesh Kumar, T. (2014). Mass loss prediction of newly developed aluminium-based alloys using artificial neural network. Neural Network World, pp.129-142.

[20] Hwang, H. (2015). A hybrid method for protein-protein interface prediction. Protein Science, 25(1), pp.159-165.

[21] Liu, C. (2017). Foreign Exchange Rates Forecasting with Convolutional Neural Network. Neural Processing Letters, 46(3), pp.1095-1119.

[22] Singh, P. (2016). Utility Study of Neural Network and Back Propagation Algorithm in The Field Of Learning And Computing Data In Mines Area. International Journal Of Engineering And Computer Science.

[23] Rafei, M. (2014). Multi-objective optimization by means of multi-dimensional MLP neural networks. Neural Network World, pp.31-56.

[24] Yaghmaei-Sabegh, S. (2017). Earthquake Ground-motion Duration Estimation by using of General Regression Neural Network. Scientia Iranica, 0(0), pp.0-0.

[25] Minami, S. (2018). Predicting Equity Price with Corporate Action Events Using LSTM-RNN. Journal of Mathematical Finance, 08(01), pp.58-63.

[26] Minami, S. (2018). Predicting Equity Price with Corporate Action Events Using LSTM-RNN. Journal of Mathematical Finance, 08(01), pp.58-63.

[27] Ma, X. (2015). Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transportation Research Part C: Emerging Technologies, 54, pp.187-197.

[28] Ong, B. (2015). Dynamically pre-trained deep recurrent neural networks using environmental monitoring data for predicting PM2.5. Neural Computing and Applications, 27(6), pp.1553-1566.

[29] Prakash, A. (2011). A Wavelet-based Neural Network Model to Predict Ambient Air Pollutants' Concentration. Environmental Modeling & Assessment, 16(5), pp.503-517.

[30] Shin, D. (2017). Deep Learning Model for Prediction Rate Improvement of Stock Price Using RNN and LSTM. The Journal of Korean Institute of Information Technology, 15(10), pp.9-16.

[31] Abdel-Nasser, M. (2017). Accurate photovoltaic power forecasting models using deep LSTM-RNN. Neural Computing and Applications.

[32] Wang, Q. (2017). Earthquake Prediction based on Spatio-Temporal Data Mining: An LSTM Network Approach. IEEE Transactions on Emerging Topics in Computing, pp.1-1.

[33] Sak, H. (2016). Modeling Dependence Dynamics of Air Pollu- tion: Pollution Risk Simulation and Prediction of PM2.5 Levels. arXiv:1602.05349.

[34] Li, X. (2017). Long short-term memory neural network for air pollutant concentration predictions: Method development and evaluation. Environmental Pollution, 231, pp.997-1004.

[35] Kingma, D. (2017). ADAM: A method for stochastic optimization. arXiv:1412.6980v9, pp.1-13.

[36] Zaremba, W. (2015). Recurrent neural network regularization. 2015 ICLR Conference.