IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 1, 2019, accepted March 11, 2019, date of publication March 28, 2019, date of current version April 13, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2908081

A Sequence-to-Sequence Air Quality Predictor Based on the n-Step Recurrent Prediction

BO LIU[1], (Senior Member, IEEE), SHUO YAN[1], JIANQIANG LI[1], (Senior Member, IEEE), GUANGZHI QU[2], (Senior Member, IEEE), YONG LI[1], JIANLEI LANG[3], AND RENTAO GU[4], (Member, IEEE)

[1]School of Software Engineering, Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China
[2]Computer Science and Engineering Department, Oakland University, Rochester, MI 48309, USA
[3]Key Laboratory of Beijing on Regional Air Pollution Control, College of Environmental & Energy Engineering, Beijing University of Technology, Beijing 100124, China
[4]Beijing Laboratory of Advanced Information Networks, School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China

Corresponding author: Jianqiang Li (lijianqiang@bjut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61702021, in part by the Beijing Natural Science Foundation under Grant 4174082 and Grant 4182040, and in part by the General Program of Science and Technology Plans of the Beijing Education Committee under Grant SQKM201710005021.

ABSTRACT Increasingly, more people are suffering from the effects of air pollution. This study took Beijing as an example and proposed an attention-based air quality predictor (AAQP) that could better protect people from air pollution. The AAQP is a seq2seq model, and it exploits historical air quality data and weather data to predict future air quality indexes. Although existing research has promoted seq2seq for air quality prediction, there are still two problems. First, the seq2seq has a slow training speed so the original RNN in the encoder was replaced with a fully connected encoder to accelerate the training process. Position embedding was also introduced to help the fully connected encoder find the sequential relationships among source sequences. Another problem is error accumulation caused by recurrent prediction. Accordingly, the n-step recurrent prediction was proposed to solve this problem. The experimental results validated that the AAQP with n-step recurrent prediction had better performance than the related arts since the error accumulation was reduced, and the training time was significantly decreased compared with the original seq2seq attention model.

INDEX TERMS Air quality, seq2seq, attention, prediction.

I. INTRODUCTION

Fossil fuels, such as coal and petroleum, have been the main energy sources since the first industrial revolution. Because abundant fossil fuels are stored in the earth and easy to exploit, they are widely used in social production by humans. Fossil fuels provide a huge amount of energy for the development of human society. Meanwhile, air pollution has become one of the thorniest problems for humans due to the combustion of fossil fuels. Combusting fossil fuels emits carbon dioxide, nitrogen oxide, sulfur dioxide and many other pollutions, which are the causes of acid rain, the greenhouse effect and other meteorological disasters. Among these air pollutants, PM2.5 is the most threatening since it causes heavy smog. The most well-known smog is the ''Great Smog of London'', which killed thousands of people. In the last decades, with rapid industrialization, air pollution in developing countries is becoming increasingly more serious than that in developed countries. As the biggest developing country, China has suffered from air pollution in recent years, especially in Beijing. Sometimes, the concentration of PM2.5 is more than 1000 $\mu g/m^3$, and the visibility is less than 1 km. Moreover, high concentration of PM2.5 may harm the respiratory system and cause lung cancer. If people do not take actions in time, high concentration PM2.5 will kill people in a short period of time. Other developing countries such as India and Mongolia are also suffering from severe air pollution. However, in terms of current technology level, there are no efficient ways to fundamentally solve heavy smog. Therefore, the best way to prevent people from being harmed by heavy smog is to predict the concentration of PM2.5 and remind people of taking actions in time.

The associate editor coordinating the review of this manuscript and approving it for publication was Zhaoxiang Zhang.

In big data era, Machine Learning (ML) has become the most important method to identify patterns of big data. With the accumulation of meteorology data and air quality data, ML became more and more popular in air quality prediction because it is easy to be implemented and has relatively high accuracy. Linear models were basic models in ML and now few researchers apply linear models to air quality predictions. Rajput *et al.* [1] used multiple linear regression (MLR) to build a model that can predict the air pollutants in India. However, in practice, air pollutants have nonlinear relationships with their influencing factors. Singh *et al.* [2] compared linear methods and nonlinear methods and found that nonlinear method could capture complex nonlinearity in air quality data. Thus nonlinear models, such as ANN [3], are more appropriate than linear models. Azid *et al.* [4] combined Principal Component Analysis (PCA) and ANN to predict the air quality in Malaysia. De Vito *et al.* [5] improved ANN with a dynamic approach. Kang [6] used data of Lanzhou and optimized ANN by genetic simulated annealing algorithm to predict air quality. Paoli *et al.* [7] used ANN to predict O_3 in Corsica. Mahajan *et al.* [8] used a geographical distance based clustering method to improve the performance of ANNs in 4 cities of Taiwan. Another nonlinear method called Support Vector Machine (SVM) [9] was also popular among researchers because it has better generalization ability than ANN. Sánchez *et al.* [10] found that SVM usually had better performance than ANN by comparing SVM with different kernels and ANN in Spain. Nieto *et al.* [11] utilized PSO-SVM based approach to predict the air quality in northern Spain. Gu *et al.* [12] extracted the sequential information of prediction by applying recurrent prediction to SVM.

Recently, deep learning has become popular because of its powerful nonlinear fitting ability. Deep learning allows researchers to easily integrate the information of different air-quality-monitor stations. Meanwhile, most researchers started to build models for predicting air quality of a future range rather than a future point. Li *et al.* [13] used a stacked autoencoder (SAE) to extract information from 12 stations and then feed extracted information into a linear regression (LR) to predict the air quality at 12 stations simultaneously. However, air quality data are sequential, so models that are good at processing sequential data such as recurrent neural network (RNN) are more powerful than SAE, ANN and SVM for air quality prediction. Ong *et al.* [14] also utilized SAE but replaced LR with RNN to give 12-hour prediction. However, RNN is hard to be trained due to its two shortcomings: exploding gradients and vanishing gradients. So, some researchers adopted Long Short-Term Memory (LSTM) to predict air quality since it was easier to be trained. Chaudhary *et al.* [15] and Pardo *et al.* [16] used simple LSTM to predict air quality for future 12 and 24 hours respectively. Zhao *et al.* [17] utilized the information of neighbor stations and LSTM to build models. Wang *et al.* [18] also used LSTM but they adopted Granger causality to choose high related stations. Zhou *et al.* [19] established a model based on LSTM and it could predict air quality of several stations. M. Kim et al [20] and KÖK *et al.* [21] found that RNN could achieve better results than ANNs and SVM. Gated Recurrent Unit (GRU) is a simplified version of LSTM and some researchers applied this method to air quality prediction. Athira *et al.* [22] compared vanilla RNN, LSTM and GRU for air quality prediction and their experiments showed that GRU had the best performance. Wang *et al.* [23] added a residual connection to GRU and LSTM and they found that GRU had better performance than LSTM. Instead of preprocessing data by RNN, Sun *et al.* [24] preprocessed data by convolution function before feeding them into predictors and their experiments turned out that GRU had better result than LSTM, ANN, SVM, random forest and MLR. Different from X. Sun *et al.*, some researchers like Du *et al.* [25], Feng *et al.* [26] and Huang *et al.* [27] adopted convolutional neural network (CNN) [28], which adapted convolution kernel to specific task, to preprocess raw data and then fed them into LSTM. Soh *et al.* [29] used the CNN to extract terrain information, e.g., a mountain between stations, and utilized LSTM and ANN to extract information from target station and its high related stations selected by cluster method. At last, they merged all the information for final prediction. Pan *et al.* [30] established a model that consisted of spatial module, deduction module and temporal module. The deduction module derived parameters of temporal module from spatial module. The temporal module, which could be CNN, LSTM or ANN models, conducted the final prediction. However, none of the above methods can treat input data and output data as sequence simultaneously. Researchers utilize either fully connected (FC) layer to predict or CNN to extract input data but all these methods lose sequential information even if CNN can only obtain fixed-length sequential information. In deep learning, the sequence-to-sequence model (seq2seq) uses RNN to extract information and predict so all the sequential information can be preserved. Reddy *et al.* [31] applied the seq2seq with LSTM to air quality prediction and proved that seq2seq was a promising method for air quality prediction. In seq2seq, the final hidden state of LSTM encoder is taken as context vector, which is decoded by LSTM decoder for final prediction, while the final hidden state of encoder cannot preserve all the useful information because some important information may be forgotten by the forget gate of LSTM. Bui *et al.* [32] takes the mean of all the encoder outputs as the context vector to solve this problem. However, different encoding time steps have different effects on decoding process, so to better address this problem, some researches tried to apply attention mechanism (AM), which was proposed by Bahdanau *et al.* [33], to seq2seq. Wang *et al.* [34] used CNN to extract the relationship among different stations and seq2seq with simplified attention for final prediction. Except for normal attention, Liang *et al.* [35] applied global attention in order to weigh the data from other sensors. Cheng *et al.* [36] adopted RNN to extract information from input sequence and AM was used to measure the influence of other stations but they used FC layer to obtain prediction results.

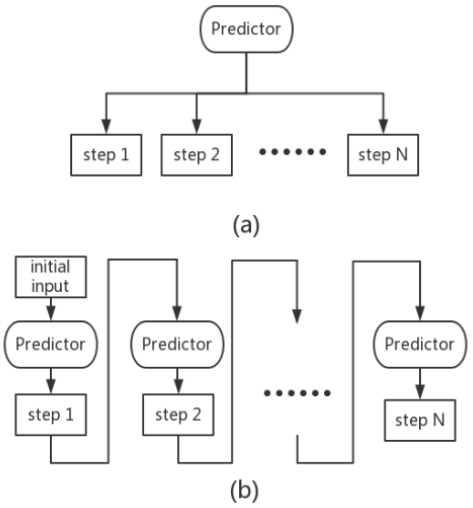


FIGURE 1. Direct prediction (a) and recurrent prediction (b). The direct prediction gives the *n* hours' predictions simultaneously. The recurrent prediction takes the output of the previous step as the input. The initial input can be the real value of the previous step.

Even when attention mechanism is applied, seq2seq has two problems. The first problem is that its training speed is too slow. In practice, models are built for each monitor station respectively. Therefore, if the model needs plenty of time for training, then it will take tremendous amount of time. The worst situation is that the models need retraining because of model deterioration caused by the variation of data distribution over time. Lots of models need to be trained simultaneously. As a result, the prediction service may have to be stopped. The seq2seq model is time-consuming because of the RNN in the encoder and decoder. The RNN runs in a step-by-step manner, which means that it cannot be paralleled. Another problem is error accumulation. Actually, seq2seq adopts recurrent prediction, which takes the output of the previous time step as the input of the current time step. Therefore, Recurrent prediction accumulates error at each time step due to the fact that predictions at each time step have error. In contrast, models that give a prediction in future period by FC layer adopted direct prediction. Direct prediction outputs the result sequence simultaneously without error accumulation but it loses the sequential information of the target sequence. The direct prediction and the recurrent prediction are shown in Figure 1. In the seq2seq, the accumulated errors also weaken the performance of the previous time steps because seq2seq is optimized according to the errors of all time steps. To accelerate the training process, we replaced the RNN in encoder with a FC layer shared by each time step. Since FC layer does not run in step-by-step manner, it can be paralleled. However, FC layer is not as powerful as RNN when processing sequential data, so position embedding is introduced to help FC layer process sequential data. To alleviate the error accumulation, we applied n-step recurrent prediction, with which the decoder gives n steps predictions directly in every time step. For instance, when $n = 3$, it means the decoder predicts the hourly air quality in future 3 hours at each time step, so when $n = 1$ it is identical with original recurrent prediction. Applying the n-step recurrent prediction to seq2seq reduces the total time steps of the decoder. Therefore, the accumulated errors are reduced and the training process is accelerated. The illustration of the n-step recurrent prediction is shown in Figure 2. The proposed method is based on attention mechanism, so it is named *n-step Attention-based Air Quality Predictor* (*n-step AAQP*).

The contributions of this paper are summarized as follows:

- The RNN encoder is replaced with the FC encoder to accelerate the training process of seq2seq.
- N-step recurrent prediction is applied to the decoder of seq2seq to improve its accuracy.
- We also compared the performance of seq2seq based on GRU and LSTM.
- Different recurrent prediction steps were applied to the AAQP and their performance were analyzed.

The rest of this paper is organized as follows. Section 2 introduces the proposed AAQP. Section 3 describes the experimental settings and discusses the results of the experiments. Section 4 gives the conclusion of this paper and future work.

II. METHODOLOGY

To better introduce the AAQP, we use a dataset $\mathbf{D} = \boldsymbol{X}, \boldsymbol{Y}$, where $\boldsymbol{X}$ is all the input sequences and $\boldsymbol{Y}$ is all the target sequences. For an input sequence $\boldsymbol{x} \in R^{S\times Q}$, its length is S and for each time step, it has Q features. For each target sequence $\boldsymbol{y} \in R^{T}$, its length is T. In practice, $\boldsymbol{y}$ can contain several targets at each time step.

A. RECURRENT NEURAL NETWORK

RNN [37] is a kind of deep neural network, whose basic structure is unit. The unit of vanilla RNN is shown in Figure 3. At each time step, the same unit is used and the input consists of the data of the current time step and the hidden state of the previous time step. The hidden state at time step s can be calculated as follows:

$$\boldsymbol{h}_s = \tanh(\boldsymbol{W} * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \mathbf{b}) \tag{1}$$

where $\boldsymbol{h}_{s-1}$ is the hidden state of previous time step, $\boldsymbol{h}_s$ is hidden state of current time step, the bracket represents the concatenation of vectors, $\boldsymbol{W}$ and $\mathbf{b}$ are the weights and biases, respectively, and they are shared at each time step. Typically, for a regression problem such as air quality prediction, the final hidden state can be used to obtain the final prediction by:

$$\boldsymbol{p} = \boldsymbol{W}_p * \boldsymbol{h}_S + \mathbf{b}_p \tag{2}$$

where $\boldsymbol{p}$ is the final prediction, $\boldsymbol{h_S}$ is the final hidden state of RNN, $\boldsymbol{W_p}$ and $\boldsymbol{b_p}$ are weights and biases.

B. LONG SHORT-TERM MEMORY UNIT

However, the vanilla RNN has two shortcomings [38]: exploding gradients and vanishing gradients. The exploding

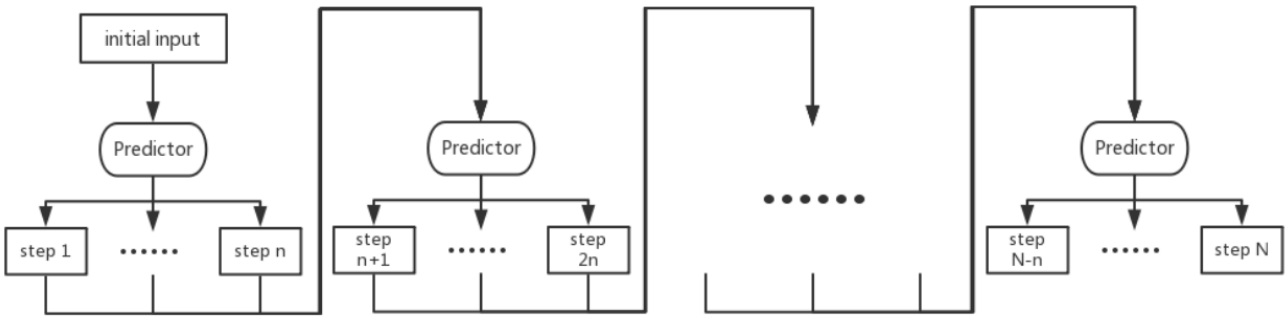


FIGURE 2. The *n*-step recurrent prediction. The predictor takes the last *n*-step as the input and outputs the *n*-step results.

gradients problem is when the norm of the gradients grows exponentially large during training. The vanishing gradients problem is when the norm of the gradient decreases exponentially to 0 during training. The former problem can be addressed using gradients clipping, which means that the gradients are constrained within a small range. The latter problem can be addressed by using the Gated Recurrent Neural Network (GRNN). The most typical GRNN is vanilla RNN whose unit is replaced by LSTM [39]. The structure of LSTM is shown in Figure 4. LSTM is comprised of a cell, an input gate, a forget gate and an output gate. The responsibility of the cell is to memorize information. The input gate controls how much new information can enter the cell, the forget gate controls how much old information should be preserved, and the output gate controls how much information of cell will be outputted. The hidden state of sth time step can be computed as follows:

$$\boldsymbol{i}_s = \sigma(\boldsymbol{W}_i * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \boldsymbol{b}_i) \tag{3}$$
$$\boldsymbol{f}_s = \sigma(\boldsymbol{W}_f * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \boldsymbol{b}_f) \tag{4}$$
$$\boldsymbol{o}_s = \sigma\,(\boldsymbol{W}_o * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \boldsymbol{b}_o) \tag{5}$$
$$\tilde{\boldsymbol{C}}_s = \tanh(\boldsymbol{W}_c * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \boldsymbol{b}_c) \tag{6}$$
$$\boldsymbol{C}_s = \boldsymbol{f}_s * \boldsymbol{C}_{s-1} + \boldsymbol{i}_s * \tilde{\boldsymbol{C}}_s \tag{7}$$
$$\boldsymbol{h}_s = \boldsymbol{o}_s * \tanh(\boldsymbol{C}_s) \tag{8}$$

where $\boldsymbol{i}_s$ is the input gate, $\boldsymbol{f}_s$ is the forget gate, and $\boldsymbol{o}_s$ is the output gate. $\tilde{\boldsymbol{C}}_s$ is a candidate cell value that represents new information, $\boldsymbol{C}_s$ represents the cell value, $\boldsymbol{h}_s$ is the current hidden state, $\boldsymbol{h}_s$ is the previous hidden state. $\boldsymbol{x}_s$ is the data of sth time step. $\boldsymbol{C}_{s-1}$ is the cell value of the previous time step, and it represents old information. $\boldsymbol{W}$ and $\boldsymbol{b}$ with different subscripts represent the different weights and biases. $\tanh(\cdot)$ is a nonlinear function, which is defined as follows:

$$tanh\,(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{9}$$

$\sigma(\cdot)$ represents the logistic function, which is defined as follows:

$$\sigma\,(x) = \frac{1}{1 + e^{-x}} \tag{10}$$

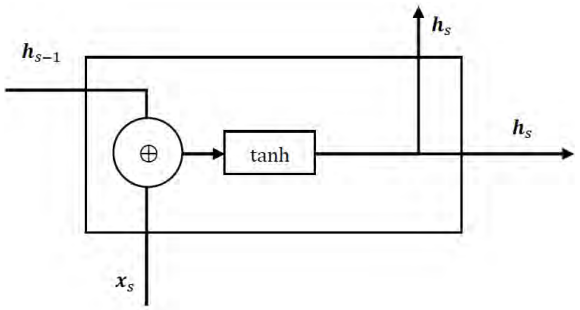


FIGURE 3. The structure of vanilla RNN unit. The ⊕ denotes concatenation.

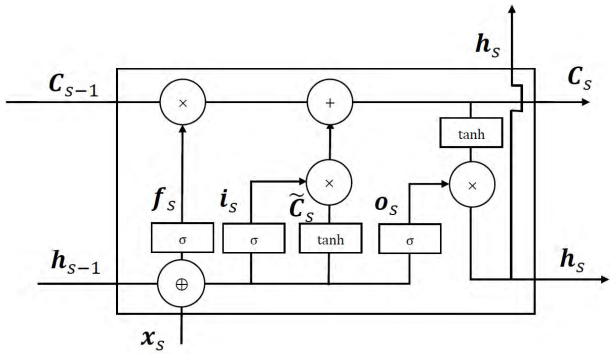


FIGURE 4. The structure of LSTM unit.

The output range of $\sigma\,(\cdot)$ is (0, 1), so the output ranges of the gates of LSTM are (0, 1) too. If the values of these gates are close to 0, almost nothing enters the cell and almost all the old information is forgotten and almost nothing will be outputted. If the values of these gates close to 1, they act the opposite behavior.

C. GATED RECURRENT UNIT

GRU [40] is a simplified version of LSTM and some researchers have implemented GRU for air quality prediction, but few researches have compared their performance of seq2seq for air quality prediction. The structure of GRU is shown in Figure 5. There are only two gates in GRU: an

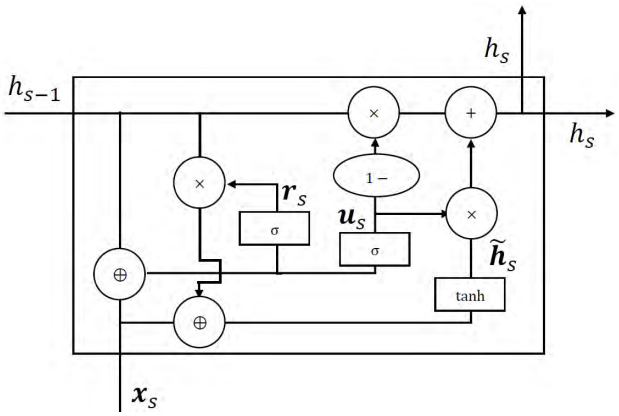


FIGURE 5. The structure of GRU unit.

update gate and a reset gate. Moreover, the cell state merges with the hidden state, so there is only a hidden state in GRU. The update gate controls how much new information enters the cell. The reset gate controls how much old information enters the cell. The hidden state $\boldsymbol{h}_s$ can be computed as follows:

$$\boldsymbol{u}_s = \sigma(\boldsymbol{W}_u * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \boldsymbol{b}_u) \tag{11}$$
$$\boldsymbol{r}_s = \sigma(\boldsymbol{W}_r * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \boldsymbol{b}_r) \tag{12}$$
$$\tilde{\boldsymbol{h}}_s = \tanh(\boldsymbol{W}_h * [\boldsymbol{r}_t * \boldsymbol{h}_{s-1}, \boldsymbol{x}_{s-1}] + \boldsymbol{b}_h) \tag{13}$$
$$\boldsymbol{h}_s = (1 - \boldsymbol{u}_s) * \boldsymbol{h}_{s-1} + \boldsymbol{u}_s * \tilde{\boldsymbol{h}}_s \tag{14}$$

where $\boldsymbol{u}_s$ is the update gate, $\boldsymbol{r}_s$ is the reset gate, and $\tilde{\boldsymbol{h}}_s$ is the candidate hidden state. The other notations are similar to LSTM. When the value of the update gate approaches 0, almost all new information is discarded. When the value of the reset gate approaches 0, old information hardly influences new information. If the values of these gates close to 1, they perform the opposite behavior.

D. SEQUENCE-TO-SEQUENCE MODEL

The seq2seq was proposed by Sutskever *et al.* [41] for machine translation and soon another advanced structure, which was used by us, was proposed by Cho *et al.* [42]. The seq2seq model has an encoder and a decoder and its structure is shown in Figure 6. It. The encoder encodes the input sequence into a context vector and then the decoder decodes the context vector into target sequence. The encoder acts totally the same as original RNN and the final hidden state $\boldsymbol{h}_S$ is the context vector. The context vector is added into the input of decoder at each time step. The input of the first time step of decoder is the observation value of air quality. Taking GRU as an example, the hidden state of decoder at tth time step is computed by:

$$\boldsymbol{u}_t = \sigma(\boldsymbol{W}_u * \left[\boldsymbol{h}_{t-1}, \boldsymbol{p}_{t-1}, \boldsymbol{h}_S\right] + \boldsymbol{b}_u) \tag{15}$$
$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_r * \left[\boldsymbol{h}_{t-1}, \boldsymbol{p}_{t-1}, \boldsymbol{h}_S\right] + \boldsymbol{b}_r) \tag{16}$$
$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}_h * \left[\boldsymbol{r}_t * \boldsymbol{h}_{t-1}, \boldsymbol{p}_{t-1}, \boldsymbol{h}_S\right] + \boldsymbol{b}_h) \tag{17}$$
$$\boldsymbol{h}_t = (1 - \boldsymbol{u}_t) * \boldsymbol{h}_{t-1} + \boldsymbol{u}_t * \tilde{\boldsymbol{h}}_t \tag{18}$$

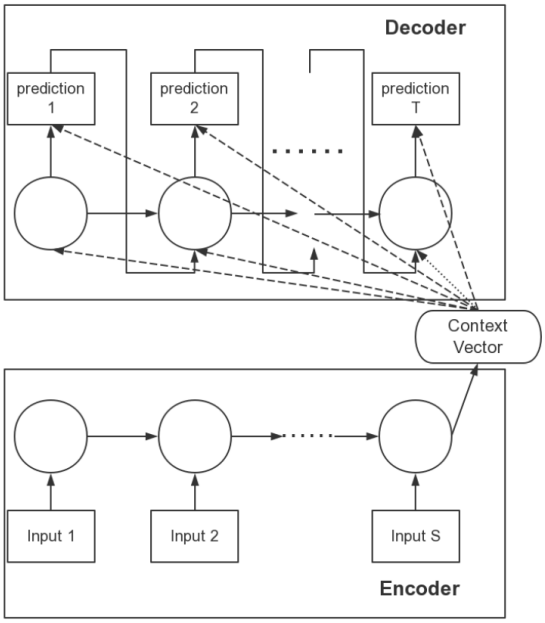


FIGURE 6. The structure of seq2seq.

where $\boldsymbol{p}_{t-1}$ is the prediction of previous time step. The prediction of current time step $\boldsymbol{p}_t$ can be computed by:

$$\boldsymbol{p}_t = \boldsymbol{W}_p * [\boldsymbol{h}_t, \boldsymbol{h}_S] + \boldsymbol{b}_p \tag{19}$$

E. N-STEP ATTENTION BASED AIR QUALITY PREDICTOR

To accelerate the training speed of seq2seq, a FC encoder replaces the RNN encoder, but the FC layer will lose the sequential information of the input sequence. To address this problem, position embedding [43] is introduced. A position embedding is a variable that has the same dimension as an input sample. Each row of the position embedding learns and stores the sequential information of each time step of input sequence. Thus, the encoder-hidden state at sth time step is computed by the following:

$$\bar{\boldsymbol{h}}_s = tanh(\boldsymbol{W}_e * (\boldsymbol{x}_s + \boldsymbol{PE}_s) + \boldsymbol{b}_e) \tag{20}$$

where $\boldsymbol{W}_e$ and $\boldsymbol{b}_e$ are the weights and biases of encoder respectively and they are shared in each time step. $\boldsymbol{PE}_s \in R^Q$ is the position embedding of sth time step. This operation for every time step can be executed simultaneously so it is more efficient. Although this combination may be not as powerful as RNN, the sequential information of the input sequence is easier to extract than that of target sequence.

The RNN decoder is preserved since the sequential information of target sequence is harder to extract than that of input sequence. However, n-step AAQP cannot obtain context vector from the last encoder-hidden state because a FC encoder replaces the RNN encoder. Therefore, the context vector is derived from AM and it is not added to the input of decoder RNN but still participates in the process of obtaining final prediction. On top of that, AM can give a greater weight to

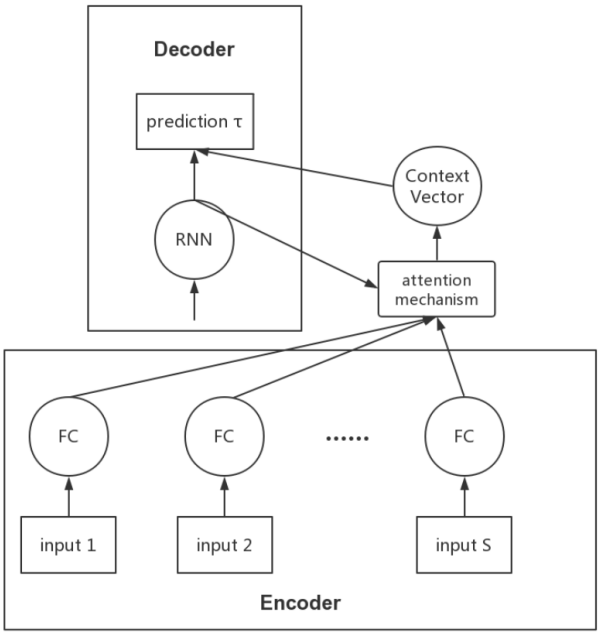


FIGURE 7. The illustration of τth time step of the *n*-step AAQP. The encoder-hidden state and decoder-hidden state are used to compute the context vector using the attention mechanism. The context vector and decoder-hidden state are used to obtain the air quality prediction.

the encoder-hidden state that has greater impact on decoder. Furthermore, *n*-step recurrent prediction is applied to decoder in order to improve the accuracy of prediction and reduce the training time. The value of *n* should be divided exactly by T, which is the length of target sequence. Thus, the number of time steps of decoder changes to T/*n*.

In this paper, we utilized the improved version of AM, which was proposed by Luong *et al.* [44]. The τth time step of decoder with AM is shown in Figure 7. Before applying AM to seq2seq, the hidden state of decoder should be computed. Taking GRU as an example, the calculation of decoder-hidden state at time step τ is shown below:

$$\boldsymbol{u}_\tau = \sigma(\boldsymbol{W}_u * \left[\boldsymbol{h}_{\tau-1}, \boldsymbol{p}_{\tau-1}\right] + \boldsymbol{b}_u) \tag{21}$$

$$\boldsymbol{r}_\tau = \sigma(\boldsymbol{W}_r * \left[\boldsymbol{h}_{\tau-1}, \boldsymbol{p}_{\tau-1}\right] + \boldsymbol{b}_r) \tag{22}$$

$$\tilde{\boldsymbol{h}}_\tau = tanh(\boldsymbol{W}_h * \left[\boldsymbol{r}_\tau * \boldsymbol{h}_{\tau-1}, \boldsymbol{p}_{\tau-1}\right] + \boldsymbol{b}_h) \tag{23}$$

$$\boldsymbol{h}_\tau = (1 - \boldsymbol{u}_\tau) * \boldsymbol{h}_{\tau-1} + \boldsymbol{u}_\tau * \tilde{\boldsymbol{h}}_\tau \tag{24}$$

To weigh the encoder-hidden state, a score is defined to measure the importance of different encoder-hidden states. The score is defined as follows:

$$\text{score}\left(\boldsymbol{h}_\tau, \bar{\boldsymbol{h}}_s\right) = \boldsymbol{h}_\tau^T \bar{\boldsymbol{h}}_s \tag{25}$$

where $\boldsymbol{h}_\tau$ is the hidden state of the decoder at time step τ, and $\bar{\boldsymbol{h}}_s$ is the hidden state of the encoder. To ensure that the summation of the weights equal 1, a softmax function is used to normalize the scores:

$$\boldsymbol{a}_\tau = \frac{exp(score\left(\boldsymbol{h}_\tau, \bar{\boldsymbol{h}}_s\right))}{\sum_{s'=1}^{S} exp(score\left(\boldsymbol{h}_\tau, \bar{\boldsymbol{h}}_{s'}\right))} \tag{26}$$

The context vector $\boldsymbol{c}_\tau$ is obtained by using the weighted average of the encoder-hidden states according to $\boldsymbol{a}_\tau$. Finally, the context vector $\boldsymbol{c}_\tau$ can be used to obtain the air quality prediction via:

$$\boldsymbol{p}_\tau = \boldsymbol{W}_p * [\boldsymbol{h}_\tau, \boldsymbol{c}_\tau] + \boldsymbol{b}_p \tag{27}$$

We used mean squared error (MSE) as loss function and since mini-batch optimization is usually used, the loss function is defined as:

$$\text{loss} = \frac{1}{M} \sum_{m=1}^{M} \sum_{\tau=1}^{T/n} ||\boldsymbol{p}_\tau^m - [\boldsymbol{y}_{n(t-1)+1}^m, \cdots \ \cdots, \boldsymbol{y}_{n*t}^m]||^2 \tag{28}$$

where M is the number of samples of a mini-batch. Finally, gradient descent is used to minimize the loss function until it converges. The complete training process of the AAQP is shown in Algorithm 1. When some data need to be predicted by trained model, execute the step (A) and (B) of Algorithm 1 for each outer loop.

III. RESULT AND DISCUSSION

The experiments were carried on a computer with I7 6770HQ CPU, 16GB RAM and GTX1060 GPU. The deep learning algorithms were implemented by tensorflow 1.8 with python 3.5. ANN and SVM were implemented by scikit-learn. All the deep learning methods used Adam as optimizer; the batch size was 512; and the learning rate was 0.001. We also clipped gradient by norm and set it to 5. Dropout was applied to all deep learning methods and its value is 0.5. The epoch of GRU and LSTM were 100. We utilized non-teacher forcing method to train seq2seq (include AAQP) so the epoch of seq2seq (include AAQP) was 1000 in order to ensure its convergence.

A. EXPERIMENT SETTING

1) DATASET

In this paper, we utilized the hourly air quality data and hourly weather data of Beijing from April 2017 to March 2018. SO_2, CO, NO_2, O_3, $PM2.5$, $PM10$ are recorded in the air quality data. The weather data include the precipitation, humidity, temperature, wind force and wind direction. The air quality data of the previous 24 hours and the weather data of the previous 24 hours were used as inputs to predict the PM2.5 of the subsequent 24 hours. The recurrent prediction needs prediction of previous time step so the seq2seq model predicted all the air pollutants because pollutants are significant information for PM2.5 prediction. Adding targets to the model can affect the training process since the model was optimized based on all targets. Additionally, too many targets may decrease the accuracy by leading to more error accumulation so the prediction targets did not contain weather factors. Furthermore, the hidden state can pass some information about pollutants and weather but we still passed pollutants to next time step in order to prevent the information from being blocked by the gates. Therefore, the shape of target sequence is 24 × 6 for one sample. However, we only

Algorithm 1 Attention-Based Air Quality Predictor

Input: dataset D, step size n
Initialize all the parameters and position embedding
For every mini-batch do:
(A) compute the hidden state of the encoder: $\bar{\boldsymbol{h}}_s = \tanh(\boldsymbol{W}_e * (\boldsymbol{x}_s + \boldsymbol{PE}_s) + \boldsymbol{b}_e)$
(B) For $\tau = 1 : T/n$:
  (a) Compute the decoder-hidden state $\boldsymbol{h}_\tau$ using equation (21)-(24) or the LSTM version of equation (21)-(24)
  (b) For every s, compute the score function: score $(\boldsymbol{h}_\tau, \bar{\boldsymbol{h}}_s) = \boldsymbol{h}_\tau^T \bar{\boldsymbol{h}}_s$
  (c) Compute the normalized weight score vector using equation (26)
  (d) Obtain the context vector $\boldsymbol{c}_\tau$ using the weighted sum of the encoder hidden states according to $\boldsymbol{a}_\tau$.
  (e) Compute the predicted value: $\boldsymbol{p}_\tau = \boldsymbol{W}_p * [\boldsymbol{h}_\tau, \boldsymbol{c}_\tau] + \boldsymbol{b}_p$
(C) For every $\boldsymbol{p}_\tau$, compute the loss function by equation (28)
(D) Perform gradient descent until convergence

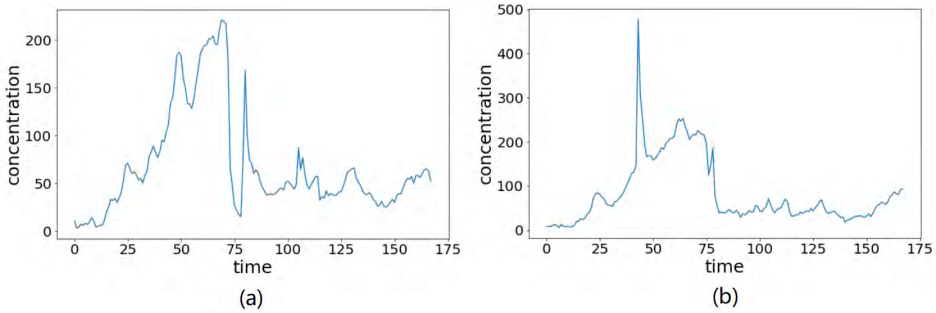


FIGURE 8. (a) PM2.5 concentration variations along time in the Olympic center Station. (b) PM2.5 concentration variations along time in Dongsi Station.

evaluate the performance of *PM2.5* prediction due to the fact that it's the main reason of haze creation. The Olympic Center station and Dongsi station are chosen as the target stations because these stations are crowded and their data are relatively complete. The *PM*2.5 has big fluctuations at Dongsi and has smaller fluctuations at Olympic Center. The *PM*2.5 concentration variations along time of the two stations is shown in Figure 8. The data from April 2017 to February 2018 are taken as the training data, and the data from March 1 2018 to March 7 2018 are the testing data.

The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

2) EVALUATION CRITERIA

In this paper, we used the mean absolute error (MAE) and R^2 as performance metrics. The MAE is defined as follows:

$$\text{MAE} = \frac{1}{n}\sum_{i=0}^{n} |O_i - P_i| \tag{29}$$

where O denotes the observation value, P denotes the predicted value, and n denotes the number of samples. The R^2 is defined as follows:

$$R^2 = 1 - \frac{\sum_{i=0}^{n} (O_i - P_i)^2}{\sum_{i=0}^{n} (O_i - \bar{O})^2} \tag{30}$$

where $\bar{O}$ is the mean of the observation values.

3) METHODS

The ANN, SVM, GRU, LSTM, seq2seq, seq2seq-mean, seq2seq-attention and n-step AAQP are applied to the dataset. The ANN, SVM, GRU and LSTM utilize the direct prediction and the others utilize the recurrent prediction. The seq2seq utilizes the hidden state of the final time step as the context vector. The seq2seq-mean utilizes the mean of the encoder-hidden state as the context vector, while seq2seq-attention and the n-step AAQP utilize the AM to compute the context vector.

B. PERFORMANCE EVALUATION

Tables 1-2 show the MAEs and R^2s of the different methods and stations. From these tables, we can conclude that the models with attention have better performance, and the recurrent prediction has better performance than the direct prediction.

At the Olympic Center station, the concentration of PM2.5 has small fluctuations, so it is easy to be predicted. The total performance, which utilized the information of all the encoder-hidden states, are similar; however, if we look closer, the MAEs of the models with attention in the previous 8 hours are smaller than those of the models without attention, and the R^2 of the models with attention in the previous 16 hours is greater than those of the models without attention. Whereas the results in the previous time steps are more trustworthy than those over the latter hours; therefore,

TABLE 1. MAE for the different methods and stations.

Stations	Methods	4	8	12	16	20	24	Total
Olympic Center	ANN	22.151	34.106	40.203	43.638	44.175	45.168	38.240
	SVM	26.283	34.553	41.099	44.923	43.267	41.710	38.639
	GRU	28.454	37.988	41.986	45.045	42.085	41.356	39.486
	LSTM	30.258	37.288	41.321	44.476	43.263	42.269	39.813
	seq2seq (GRU)	21.322	33.593	36.211	40.313	44.061	46.528	37.004
	seq2seq (LSTM)	19.020	29.515	36.705	44.334	45.254	49.002	37.495
	seq2seq-mean (GRU)	18.617	28.606	32.200	37.050	39.049	43.230	**33.109**
	seq2seq-mean (LSTM)	21.823	32.631	36.194	35.448	**36.278**	41.238	33.935
	seq2seq-attention (GRU)	19.118	**28.572**	**31.965**	**32.758**	40.618	46.714	33.308
	seq2seq-attention (LSTM)	18.525	31.362	36.182	37.777	36.912	40.120	33.480
	1-step AAQP (GRU)	**17.807**	30.994	35.618	37.501	39.702	39.702	33.554
	1-step AAQP (LSTM)	20.424	29.941	35.727	38.472	39.690	**38.128**	33.730
Dongsi	ANN	40.536	52.584	55.542	57.185	67.490	61.071	55.735
	SVM	56.433	67.628	67.730	68.125	67.592	66.307	65.636
	GRU	34.964	45.855	46.763	45.666	48.732	54.617	46.099
	LSTM	33.022	43.038	49.595	53.087	55.899	55.070	48.285
	seq2seq (GRU)	28.475	42.467	46.942	55.174	60.444	57.218	48.453
	seq2seq (LSTM)	27.465	40.937	45.808	50.628	54.171	59.524	46.103
	seq2seq-mean (GRU)	26.847	38.751	48.742	51.130	50.880	55.071	46.237
	seq2seq-mean (LSTM)	26.279	44.702	58.632	66.136	67.909	63.702	54.560
	seq2seq-attention (GRU)	26.890	42.772	50.419	47.524	49.099	54.955	45.276
	seq2seq-attention (LSTM)	24.899	37.859	48.579	57.516	58.465	51.324	46.440
	1-step AAQP (GRU)	**24.323**	40.239	**44.488**	**46.700**	**46.850**	**46.208**	**41.468**
	1-step AAQP (LSTM)	25.199	**33.657**	41.330	48.299	50.168	53.280	41.989

TABLE 2. R^2 for the different methods and stations.

Stations	Methods	4	8	12	16	20	24	Total
Olympic Center	ANN	0.657	0.280	0.081	-0.101	-0.112	-0.291	0.086
	SVM	0.609	0.274	0.045	-0.121	-0.097	-0.077	0.105
	GRU	0.419	0.069	-0.339	-0.622	-0.378	-0.341	-0.193
	LSTM	0.276	0.035	-0.171	-0.400	-0.439	-0.393	-0.177
	seq2seq (GRU)	0.607	0.143	-0.026	-0.342	-0.500	-0.496	-0.102
	seq2seq (LSTM)	0.648	0.431	0.195	-0.253	-0.369	-0.393	0.043
	seq2seq-mean (GRU)	0.692	0.454	0.395	0.178	-0.035	-0.334	0.225
	seq2seq-mean (LSTM)	0.680	0.388	0.353	0.322	-0.199	-0.669	0.246
	seq2seq-attention (GRU)	0.692	**0.475**	**0.437**	**0.401**	-0.051	-0.433	**0.253**
	seq2seq-attention (LSTM)	**0.729**	0.403	0.275	0.094	0.112	-0.101	0.252
	1-step AAQP (GRU)	0.710	0.251	0.169	0.158	0.031	-0.036	0.213
	1-step AAQP (LSTM)	0.678	0.358	0.260	0.122	**0.009**	**-0.007**	0.237
Dongsi	ANN	0.444	0.158	0.058	-0.016	-0.341	-0.250	0.008
	SVM	0.111	-0.186	-0.226	-0.266	-0.254	-0.197	-0.170
	GRU	0.411	0.247	0.178	0.104	-0.028	-0.100	0.143
	LSTM	0.340	0.094	-0.041	-0.170	-0.372	-0.345	-0.071
	seq2seq (GRU)	0.475	0.029	0.036	-0.203	-0.290	-0.161	-0.018
	seq2seq (LSTM)	0.507	0.273	0.078	-0.391	-0.383	-0.226	-0.080
	seq2seq-mean (GRU)	0.649	0.431	-0.117	-0.188	-0.095	-0.290	0.065
	seq2seq-mean (LSTM)	0.563	0.107	-0.362	-0.535	-0.617	-0.568	-0.235
	seq2seq-attention (GRU)	0.574	0.224	-0.008	0.034	-0.094	-0.288	0.073
	seq2seq-attention (LSTM)	0.623	0.354	0.029	-0.320	-0.321	-0.044	0.053
	1-step AAQP (GRU)	0.650	0.298	0.195	**0.116**	**0.056**	**0.056**	**0.228**
	1-step AAQP (LSTM)	**0.663**	**0.492**	**0.308**	0.065	-0.085	-0.229	0.202

the attention models can provide better results. The proposed AAQP (GRU) method has similar performance to the original seq2seq model with attention, although the AAQP is simplified, and so the simplification is reasonable.

At Dongsi station, the concentration of PM2.5 has larger fluctuations than that at the Olympic Center station, which means it is hard for the models to get an accurate prediction. Attention models have the best performance in both the previous time steps and the latter time steps. The AAQP (LSTM) has the best performance in the previous 12 hours, and so it is the best model for Dongsi station. At Dongsi station, the AAQP gets a better result than the original seq2seq attention model, and thus this is an appropriate simplification for the air quality prediction.

TABLE 3. MAE for the different steps of AAQP.

Stations	method	steps	4	8	12	16	20	24	Total
Olympic Center	AAQP (GRU)	1	18.683	28.376	**34.871**	40.111	41.928	44.462	34.738
		2	19.726	29.897	35.508	37.477	**36.627**	**34.345**	**32.263**
		3	20.647	31.239	36.024	**36.972**	37.073	40.031	33.664
		4	20.312	31.535	37.688	42.783	40.980	42.153	35.908
		6	18.174	29.334	37.626	38.869	37.710	37.849	33.260
		12	**15.599**	**28.074**	35.422	42.700	43.815	43.301	34.818
	AAQP (LSTM)	1	26.387	39.066	44.554	47.746	47.442	47.671	35.256
		2	19.791	27.550	**33.184**	37.312	**36.397**	**35.326**	31.593
		3	20.108	32.321	36.626	38.549	39.135	35.787	33.755
		4	18.404	29.196	33.612	37.747	40.018	38.772	32.958
		6	17.647	**27.262**	33.900	**35.870**	36.424	35.870	**31.162**
		12	**15.534**	28.407	36.326	40.540	41.713	42.155	34.112
	Best		18.525	28.376	**30.302**	**31.588**	36.912	40.120	32.848
Dongsi	AAQP (GRU)	1	24.323	40.239	44.488	46.700	46.850	46.208	41.468
		2	26.915	38.549	43.926	44.245	44.852	49.369	41.309
		3	25.025	33.935	**40.249**	47.011	45.812	47.399	39.905
		4	21.647	34.960	44.379	46.418	48.048	46.678	40.355
		6	23.540	36.786	41.818	46.550	**44.205**	**44.179**	39.513
		12	**19.910**	**33.772**	40.547	**44.046**	45.079	47.629	**38.497**
	AAQP (LSTM)	1	25.199	33.657	41.330	48.299	50.168	53.280	41.989
		2	25.208	33.967	41.983	43.682	43.435	46.722	39.166
		3	24.437	36.138	42.463	44.385	43.906	**43.909**	39.206
		4	21.765	33.081	**37.401**	**41.670**	43.834	46.689	**37.407**
		6	22.199	32.742	39.940	41.997	**43.266**	45.328	37.579
		12	**20.513**	**31.527**	39.161	42.089	45.999	49.893	38.197
	Best		24.323	37.859	44.488	46.700	46.850	46.208	41.468

Although GRU and LSTM extract the sequential information from historical data, their performance is not always better than ANN and SVM. This fact implies that the FC encoder can extract sequential information, but when approximating a tough curve, the ANN and SVM are not as powerful as GRU and LSTM. The PE helps the AAQP to extract the sequential information so the AAQP has better performance than the other seq2seq models at Dongsi station. The seq2seq models have better performance than LSTM and GRU, which indicates that recurrent prediction is better than direct prediction. Therefore, replacing RNN with FC layer and preserving RNN decoder are good choices.

C. AAQP WITH DIFFERENT STEPS

The performance of the AAQP with different steps is shown in Tables 3-4. The red numbers represent the best result for a station, bold numbers represent the best results for a method, and the row named "Best" is the best results of the corresponding columns in Tables 1-2. In general, the 12-step AAQP is the best since it always obtains the best performance over the previous 8 hours. Almost all the best results are obtained by the n-step AAQP, and so the n-step recurrent prediction significantly helps the AAQP to improve. It is noteworthy that there is no strict monotonic relationship between n and the performance, although a bigger n usually performs well over the previous 8 hours. After 8 hours, a smaller n is more advantageous at Olympic Center station, and a bigger n is more advantageous at Dongsi Station. 12-step AAQPs always get the best performance at first 4 hours. In these cases, the weights and bias of output layer only need to adapt 2 outputs, but in other cases, they need to adapt to many outputs. Therefore, 12-step AAQPs are easier to achieve higher accuracy at first 4 hours. These tables also show us that 1-step AAQPs have the worst result at latter hours in most cases. The total performance of 1-step AAQPs are always the worst or very close to the worst. These 2 phenomena indicate that n-step recurrent prediction is helpful to reduce the error accumulation if n is chosen appropriately. At Olympic Center, models with a relatively small n have better total performance. The reason is that the target of Olympic Center station has small fluctuations, every time step has a smaller error, the accumulated error is small. Whereas, a big n will decrease the performance of latter hours and lead to poor total performance. For example, the performance of 12-step AAQPs are significantly different between previous 12 hours and latter 12 hours. The target of Dongsi station has large fluctuations, and the error of each time step is big, so a small n will accumulate more error. In contrast, a big n will not accumulate too much error. For instance, the performance of 12-step AAQPs has no significant differences with previous 12 hours and latter 12 hours but decreases naturally over time. In summary, if the target has large fluctuations, a big n should be chosen for total performance; otherwise, a small n should be selected. The 12-step AAQP (LSTM) has the best result at Olympic Center because it has the best performance over the previous 4 hours, and the performance over the past 5-12 hours is very close to the best. The 12-step AAQP (GRU) achieves the best result at Dongsi station, it is

TABLE 4. R^2 for the different steps of AAQP.

Stations	method	steps	4	8	12	16	20	24	Total
Olympic Center	AAQP (GRU)	1	0.703	**0.444**	**0.275**	-0.022	-0.211	-0.351	0.139
		2	0.682	0.370	0.198	0.126	0.152	**0.233**	**0.294**
		3	0.663	0.354	0.221	**0.173**	0.090	-0.103	0.233
		4	0.657	0.282	0.132	-0.090	-0.075	-0.229	0.112
		6	0.694	0.391	0.106	0.105	**0.167**	0.178	0.273
		12	**0.718**	0.402	0.240	-0.050	-0.117	-0.138	0.175
	AAQP (LSTM)	1	0.624	0.327	0.171	0.078	**0.121**	0.070	0.049
		2	0.688	**0.438**	0.235	0.027	0.049	0.124	0.260
		3	0.680	0.244	0.203	0.089	0.039	**0.234**	0.248
		4	0.712	0.377	**0.261**	0.064	-0.094	-0.016	0.217
		6	0.716	0.427	0.230	**0.217**	0.093	0.083	**0.294**
		12	**0.739**	0.433	0.232	-0.002	-0.137	-0.155	0.185
	Best		0.729	**0.475**	**0.437**	**0.401**	0.121	0.070	0.253
Dongsi	AAQP (GRU)	1	0.650	0.298	0.195	0.116	0.056	0.056	0.228
		2	0.651	0.376	0.260	0.171	**0.196**	0.018	0.279
		3	0.668	0.453	0.294	0.001	0.085	0.073	0.262
		4	**0.708**	0.444	0.211	0.001	-0.034	0.027	0.226
		6	0.690	0.415	0.219	0.089	0.116	**0.081**	0.269
		12	**0.708**	**0.502**	**0.337**	**0.219**	0.098	-0.016	**0.308**
	AAQP (LSTM)	1	0.663	0.492	0.308	0.065	-0.085	-0.229	0.202
		2	0.635	0.412	0.228	0.150	0.135	-0.003	0.259
		3	0.665	0.360	0.242	0.121	**0.170**	**0.188**	0.291
		4	**0.687**	0.424	**0.350**	0.233	0.091	0.022	0.301
		6	0.678	0.482	0.284	0.190	0.156	0.102	0.315
		12	0.685	**0.513**	0.309	**0.240**	0.150	0.001	**0.316**
	Best		0.650	0.492	0.308	0.116	0.056	0.056	0.228

TABLE 5. Training and prediction time.

method	Training time (s)	Prediction time(s)
seq2seq (GRU)	392.832	2.378
seq2seq (LSTM)	489.456	3.944
seq2seq-mean (GRU)	419.904	1.711
seq2seq-mean (LSTM)	776.880	2.013
seq2seq-attention (GRU)	1577.952	5.520
seq2seq-attention (LSTM)	2112.624	5.673
1-step AAQP (GRU)	1239.408	3.576
1-step AAQP (LSTM)	1080.156	4.405
2-step AAQP (GRU)	843.696	1.333
2-step AAQP (LSTM)	890.496	1.163
3-step AAQP (GRU)	663.699	1.295
3-step AAQP (LSTM)	645.264	0.936
4-step AAQP (GRU)	519.735	1.160
4-step AAQP (LSTM)	519.840	0.756
6-step AAQP (GRU)	430.272	0.679
6-step AAQP (LSTM)	435.168	0.612
12-step AAQP (GRU)	338.544	0.888
12-step AAQP (LSTM)	341.136	0.415

the best model over the previous 4 hours, and its performance over the previous 5-16 hours is very close to the best.

D. TIME

Table 5 shows the training time and prediction time of different methods. AM is a time-consuming operation due to the fact it needs to be operated every time step at decode stage, so it needs 4 times the training time compared to original seq2seq. The simplification of the encoder decreases the training time, but it is still much longer than that of the original seq2seq. Applying the n-step recurrent prediction significantly reduces the training time because of the reduction of the time steps in the decoder. A bigger n results in a shorter training time. When $n = 6$, the training time is very close to the original seq2seq, and when $n = 12$, the training time is shorter than the original seq2seq. The prediction time of every model are very short and have no significant influence on real-time predictions.

E. VISUALIZATION

Figures 9-10 visualize the results at Olympic Center and Dongsi station, and every subplot shows the different predictions of different hours. Figure 9 is the visualization results of the 12-step AAQP (LSTM), which has the best performance at Olympic Center. At the first hour, the prediction is almost equal to the real values. At the second hour the prediction starts to lag behind the real values, but the lags are small. Starting at the third hour, the lags become serious, and the result is not trustworthy. Figure 10 is the visualization of the 12-step AAQP (GRU), which has the best performance at Dongsi station. In the first hour, the prediction is very close to the real values. At the second hour, small lags appear. Begin with the third hour, the lags become obvious.

Predictions of all methods lag behind the observation values starting from the second hour, which is because the accuracy decreases as the predictor predicts the PM2.5 at the following hours. However, we had found two fairly good modifications for this problem. One is adding layers, another is using weather forecast. Therefore, we used 2 FC layers in encoder and 2 RNN layers in decoder. The 4 layers had

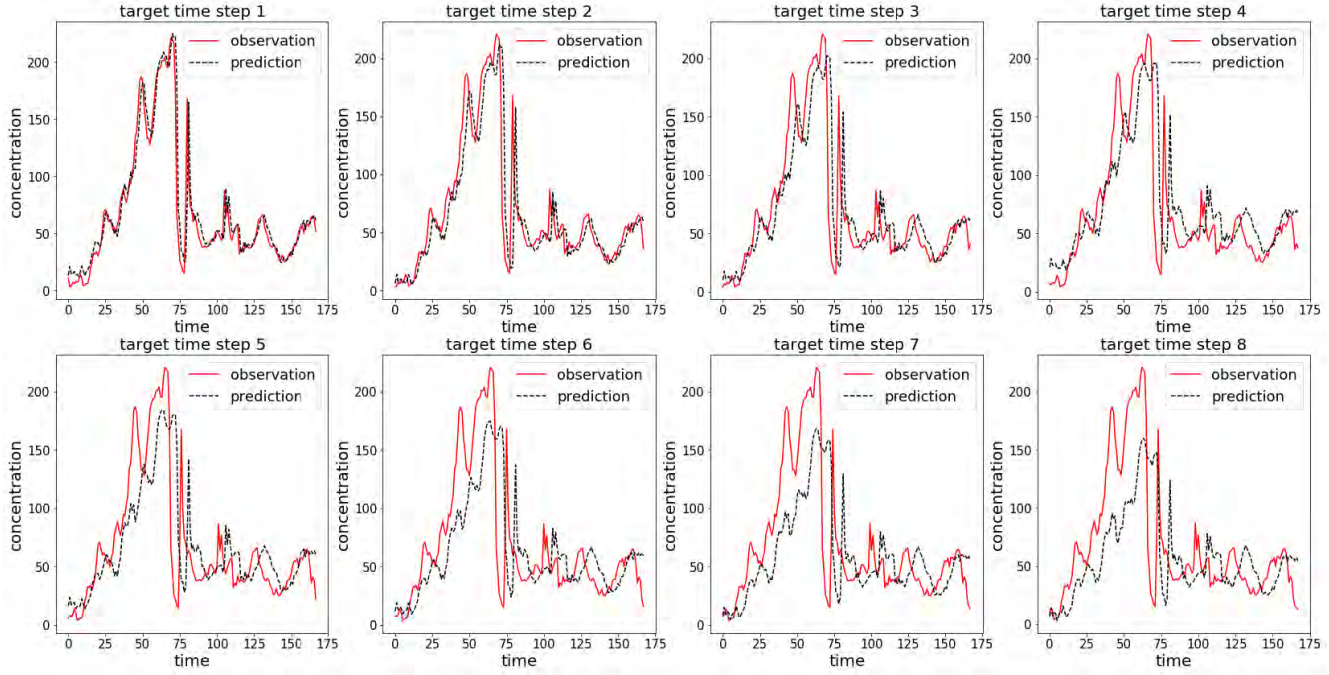


FIGURE 9. The visualization of the predictions at Olympic Center station.

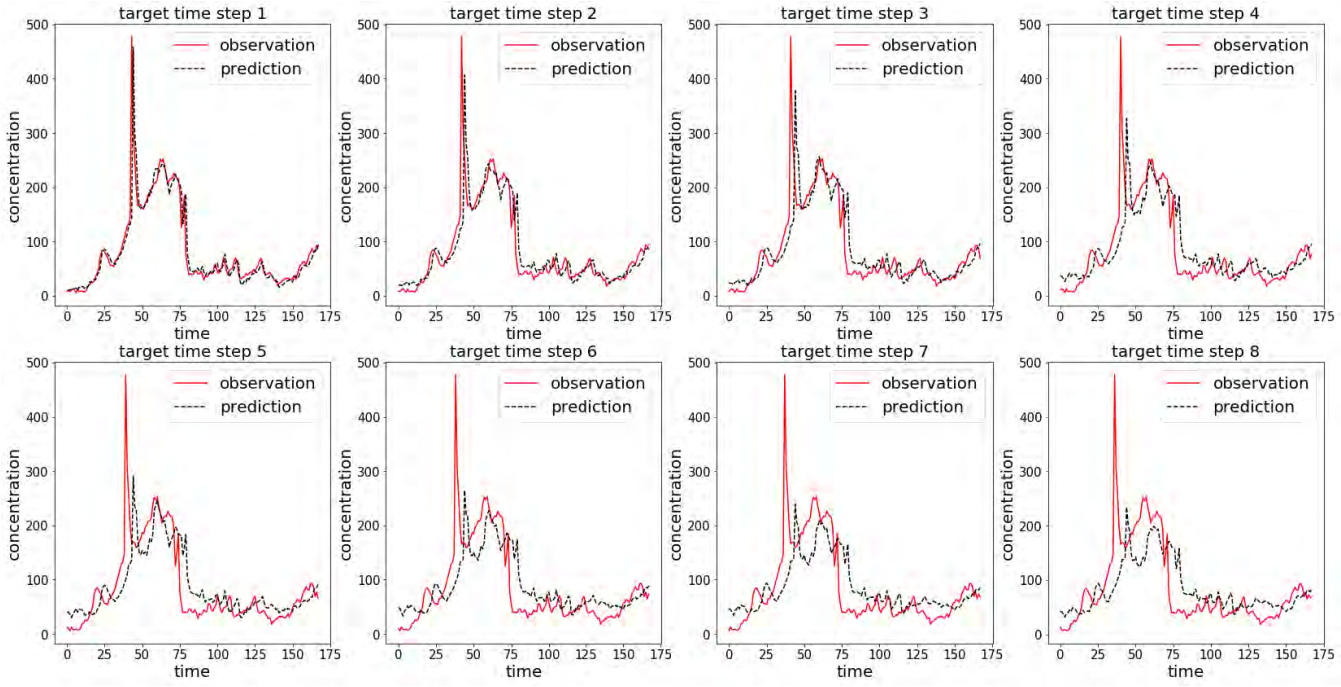


FIGURE 10. The visualization of the predictions at Dongsi station.

64 units respectively. Additionally, we added hourly weather forecast data as inputs. Although we didn't have weather forecast data, it is reasonable to use mock data which are true data of future even if forecasting data are not as accurate as true data because neural networks can adapt to the error of forecasting data. Therefore, taking GRU as an example, the computation of decoder-hidden state is changed to:

$$\boldsymbol{u}_\tau = \sigma(\boldsymbol{W}_u * \left[\boldsymbol{h}_{\tau-1}, \boldsymbol{wf}_\tau, \boldsymbol{p}_{\tau-1}\right] + \boldsymbol{b}_u) \tag{31}$$
$$\boldsymbol{r}_\tau = \sigma(\boldsymbol{W}_r * \left[\boldsymbol{h}_{\tau-1}, \boldsymbol{wf}_\tau, \boldsymbol{p}_{\tau-1}\right] + \boldsymbol{b}_r) \tag{32}$$
$$\tilde{\boldsymbol{h}}_\tau = \tanh(\boldsymbol{W}_h * \left[\boldsymbol{r}_\tau * \boldsymbol{h}_{\tau-1}, \boldsymbol{wf}_\tau, \boldsymbol{p}_{\tau-1}\right] + \boldsymbol{b}_h) \tag{33}$$
$$\boldsymbol{h}_\tau = (1 - \boldsymbol{u}_\tau) * \boldsymbol{h}_{\tau-1} + \boldsymbol{u}_\tau * \tilde{\boldsymbol{h}}_\tau \tag{34}$$

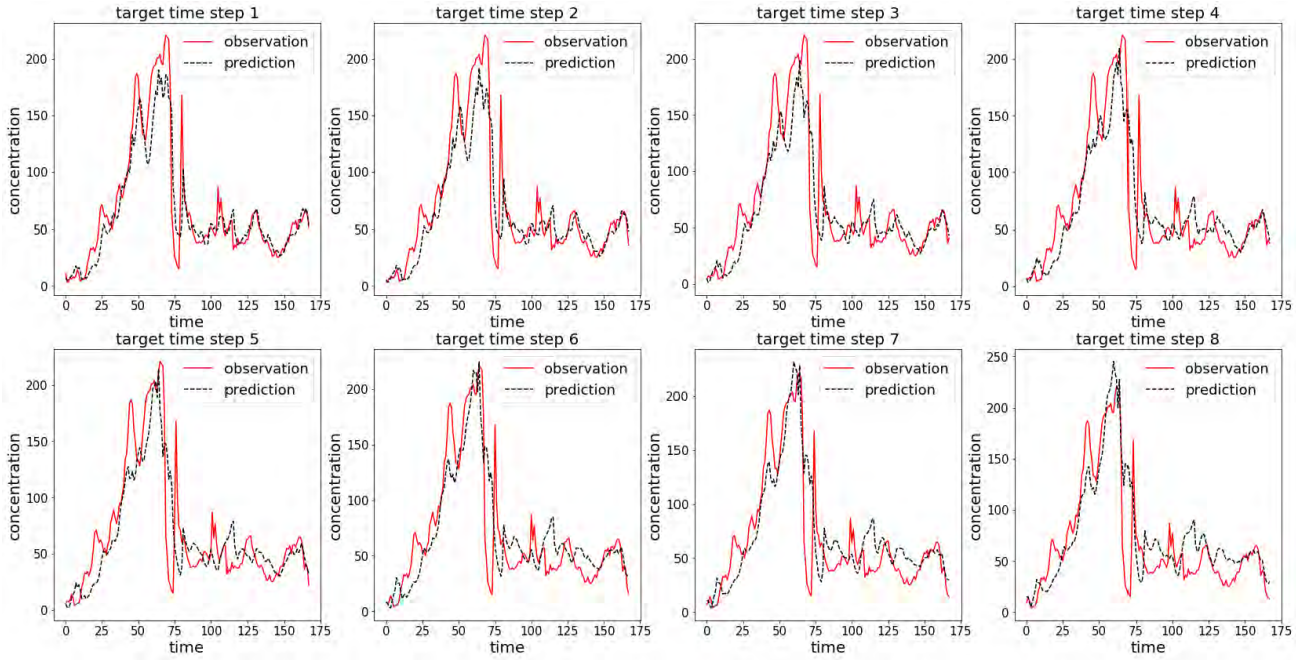


FIGURE 11. The visualization of the predictions at Olympic Center station with two modifications.

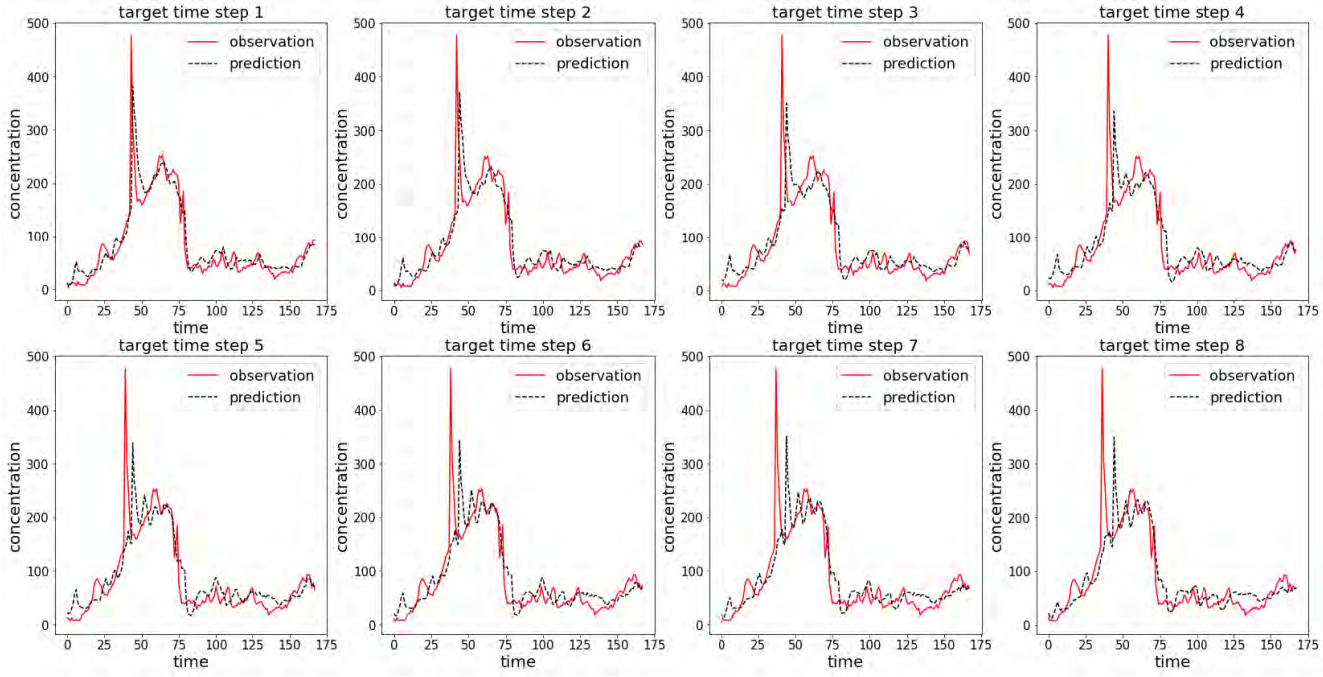


FIGURE 12. The visualization of the predictions at Olympic Center station with two modifications.

where $\boldsymbol{wf}_\tau$ is weather forecast data of current time step τ. 12-step AAQP was used to show the effect of these modifications. The result is shown in Table 5-6 and the methods end with "M" are methods with the two modifications. We can see that the total MAE and R^2 are significantly improved by the two modifications. The MAEs of first 4 hours slightly decrease while the R^2s of first 4 hours increase which means that the trend of prediction became more similar to the observation value. From the tables we can conclude that the 12-step AAQP (GRU) has better performance than 12-step AAQP (LSTM) with two modifications. To show the details, the visualizations of 12-step AAQP (GRU) of two stations are depicted in Figures 11-12. These figures show that the lags have been significantly

TABLE 6. R^2 for the different methods and stations.

Stations	Methods	4	8	12	16	20	24	Total
Olympic Center	12-step AAQP (GRU)	15.599	28.074	35.422	42.700	43.815	43.301	34.818
	12-step AAQP (LSTM)	**15.534**	28.407	36.326	40.540	41.713	42.155	34.112
	12-step AAQP (GRU) M	16.102	**20.289**	23.678	**24.617**	**23.894**	**27.838**	**22.736**
	12-step AAQP (LSTM) M	16.644	21.722	**22.918**	25.195	27.989	29.921	24.065
Dong Si	12-step AAQP (GRU)	**19.910**	33.772	40.547	44.046	45.079	47.629	38.497
	12-step AAQP (LSTM)	20.513	31.527	39.161	42.089	45.999	49.893	38.197
	12-step AAQP (GRU) M	21.589	**23.116**	**25.865**	**29.519**	**32.221**	**38.532**	**28.473**
	12-step AAQP (LSTM) M	20.862	27.607	29.493	30.677	33.745	39.668	30.342

TABLE 7. R^2 for the different methods and stations.

Stations	Methods	4	8	12	16	20	24	Total
Olympic Center	12-step AAQP (GRU)	0.718	0.402	0.240	-0.050	-0.117	-0.138	0.175
	12-step AAQP (LSTM)	0.739	0.433	0.232	-0.002	-0.137	-0.155	0.185
	12-step AAQP (GRU) M	**0.785**	**0.704**	0.633	**0.625**	**0.612**	**0.521**	**0.647**
	12-step AAQP (LSTM) M	0.775	0.691	**0.647**	0.557	0.512	0.456	0.607
Dong Si	12-step AAQP (GRU)	0.708	0.502	0.337	0.219	0.098	-0.016	0.308
	12-step AAQP (LSTM)	0.685	0.513	0.309	0.240	0.150	0.001	0.316
	12-step AAQP (GRU) M	**0.739**	**0.702**	**0.660**	**0.582**	**0.551**	**0.433**	**0.611**
	12-step AAQP (LSTM) M	0.710	0.592	0.597	0.548	0.492	0.409	0.558

reduced except for large sudden changes. Moreover, at most of times, the model can produce trustworthy predictions. Thus, with sacrificing a little accuracy at first 4 hours, the lags decreased obviously.

According to the above experimental results, it seems that the predictors without two modifications did not approximate the target but the prediction of first hour. Then, we can derive an assumption of the apperance of lag. The lags appeared since the predictor found that approximating the target could not achieve high accuracy so the predictor found a strategy to approximate the first output meanwhile make the first output as accurate as possible. As a result, the first output became extremely accurate and other outputs imitated the first output. Because these outputs shared many parameters, it was easy to approximate the first output. In this way, the loss is smaller than approximating the target. By contrast, the weather forecast data gave more information to the latter predictions so they could achieve higher accuracy than imitation first output when they approximated the target. More layers also helped the predictor improve its accuracy. Besides, the predictor no longer focused on first output because other outputs did not need to imitate it. Except for our research, other researches [7], [20], [27], [45] that visualized their predictions also have the same problem regardless of the deep learning or the machine learning. Particularly, Du *et al.* [25] showed that their predictions hadn't lag with one output but the lag appeared when multioutput is used. Therefore, the lag problem is common in air quality prediction.

Our conclusion is that, the loss function needs to be modified, so that the predictor can be restricted from approximating first prediction especially for multioutput scenario.

IV. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed the *n*-step AAQP, which is an attention*n*-based seq2seq model, for air quality prediction. The *n*-step AAQP had better performance than seq2seq models. To accelerate the training process of seq2seq with attention, a FC encoder replaced the RNN encoder of seq2seq. In addition, position embedding was introduced to help the FC encoder extract the sequential information. Moreover, the performance of the AAQP was close to the seq2seq with attention at the Olympic Center station and is even better at Dongsi station. To overcome the shortcomings of the accumulated errors as the time step grows, the *n*-step recurrent prediction was applied. Through *n*-step recurrent prediction, the performance of the AAQP was significantly improved. In addition, the training of seq2seq was further accelerated. The two promotions make seq2seq have more accurate predictions and higher training speed. Particularly, the AAQP can give a trustworthy alert 2 hours in advance before sudden air pollution strikes. Additionally, the weather forecast data are essential to improve the accuracy of air quality prediction.

This work is focused on temporal attention and utilizes single station information, but in practice, there are relationships between different stations. Therefore, in the future, we will work on spatial attention to further improve the performance of the AAQP. In addition, we will find new loss function to solve the lag problem and collect some weather forecast data to improve the accuracy.

REFERENCES

[1] T. S. Rajput and N. Sharma, "Multivariate regression analysis of air quality index for Hyderabad city: Forecasting model with hourly frequency," *Int. J. Appl. Res.*, vol. 3, no. 8, pp. 443–447, 2017.

[2] K. P. Singh, S. Gupta, A. Kumar, and S. P. Shukla, "Linear and nonlinear modeling approaches for urban air quality prediction," *Sci. Total Environ.*, vol. 426, pp. 244–255, Jun. 2012.

[3] S. Dreiseitl and L. Ohno-Machado, "Logistic regression and artificial neural network classification models: A methodology review," *J. Biomed. Inform.*, vol. 35, nos. 5–6, pp. 352–359, 2002.

[4] A. Azid *et al.*, "Prediction of the level of air pollution using principal component analysis and artificial neural network techniques: A case study in Malaysia," *Water, Air, Soil Pollution*, vol. 225, no. 8, p. 2063, 2014.

[5] S. De Vito *et al.*, "Dynamic multivariate regression for on-field calibration of high speed air quality chemical multi-sensor systems," in *Proc. AISEM Annu. Conf.*, Feb. 2015, pp. 1–3.

[6] Z. Kang and Z. Qu, "Application of BP neural network optimized by genetic simulated annealing algorithm to prediction of air quality index in Lanzhou," in *Proc. IEEE Comput. Intell. Appl. (ICCIA)*, Sep. 2017, pp. 155–160.

[7] C. Paoli, G. Notton, M. Nivet, M. Padovani, and J. Savelli, "A neural network model forecasting for prediction of hourly ozone concentration in Corsica," in *Proc. Environ. Elect. Eng. (EEEIC)*, May 2011, pp. 1–4.

[8] S. Mahajan, H. M. Liu, T. C. Tsai, and L. J. Chen, "Improving the accuracy and efficiency of PM2.5 forecast service using cluster-based hybrid neural network model," *IEEE Access*, vol. 6, pp. 19193–19204, 2018.

[9] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[10] A. S. Sánchez, P. J. G. Nieto, P. R. Fernández, J. J. del Coz Díaz, and F. J. Iglesias-Rodríguez, "Application of an SVM-based regression model to the air quality study at local scale in the Avilés urban area (Spain)," *Math. Comput. Model.*, vol. 54, nos. 5–6, pp. 1453–1466, 2011.

[11] P. J. G. Nieto, E. Garcia-Gonzalo, F. S. Lasheras, and F. J. de Cos Juez, "Hybrid PSO–SVM-based method for forecasting of the remaining useful life for aircraft engines and evaluation of its reliability," *Rel. Eng. Syst. Saf.*, vol. 138, pp. 219–231, Jun. 2015.

[12] K. Gu, J. Qiao, and W. Lin, "Recurrent air quality predictor based on meteorology- and pollution-related factors," *IEEE Trans. Ind. Informat.*, vol. 14, no. 9, pp. 3946–3955, Sep. 2018.

[13] X. Li, L. Peng, Y. Hu, J. Shao, and T. Chi, "Deep learning architecture for air quality predictions," *Environ. Sci. Pollut. Res.*, vol. 23, no. 22, pp. 22408–22417, 2016.

[14] B. T. Ong, K. Sugiura, and K. Zettsu, "Dynamic pre-training of deep recurrent neural networks for predicting environmental monitoring data," in *Proc. IEEE Big Data*, Oct. 2014, pp. 760–765.

[15] V. Chaudhary, A. Deshbhratar, V. Kumar, and D. Paul. *Time Series Based LSTM Model to Predict Air Pollutant's Concentration for Prominent Cities in India*. Accessed: Jan. 25, 2019. [Online]. Available: http://philippe-fournier-viger.com/utility_mining_workshop_2018/PAPER1.pdf

[16] E. Pardo and N. Malpica, "Air quality forecasting in Madrid using long short-term memory networks," in *Proc. Int. Work-Conf. Interplay Between Natural Artif. Comput.*, 2017, pp. 232–239.

[17] J. Zhao, F. Deng, Y. Cai, and J. Chen, "Long short-term memory-fully connected (LSTM-FC) neural network for PM2.5 concentration prediction," *Chemosphere*, vol. 220, pp. 486–492, Apr. 2019.

[18] J. Wang and G. Song, "A deep spatial-temporal ensemble model for air quality prediction," *Neurocomputing*, vol. 314, pp. 198–206, Nov. 2018.

[19] Y. Zhou, F.-J. Chang, L.-C. Chang, I.-F. Kao, and Y.-S. Wang, "Explore a deep learning multi-output neural network for regional multi-step-ahead air quality forecasts," *J. Clean Prod.*, vol. 209, pp. 134–145, Feb. 2019.

[20] M. Kim, Y. Kim, S. Sung, and C. Yoo, "Data-driven prediction model of indoor air quality by the preprocessed recurrent neural networks," in *Proc. IEEE ICCAS-SICE*, Aug. 2009, pp. 1688–1692.

[21] İ. Kök, M. U. Şimşek, and S. Özdemir, "A deep learning model for air quality prediction in smart cities," in *Proc. IEEE Big Data*, Dec. 2017, pp. 1983–1990.

[22] V. Athira, P. Geetha, R. Vinayakumar, and K. P. Soman, "DeepAirNet: Applying recurrent networks for air quality prediction," *Procedia Comput. Sci.*, vol. 132, pp. 1394–1403, Dec. 2018.

[23] B. Wang, Z. Yan, J. Lu, G. Zhang, and T. Li, "Deep multi-task learning for air quality prediction," in *Proc. Int. Conf. Neural Inf. Process.*, 2018, pp. 93–103.

[24] X. Sun, W. Xu, and H. Jiang, "Spatial-temporal prediction of air quality based on recurrent neural networks," in *Proc. 52nd Hawaii Int. Conf. Syst. Sci.*, 2019, pp. 1–10.

[25] S. Du, T. Li, Y. Yang, and S.-J. Horng. (2018). "Deep air quality forecasting using hybrid deep learning framework." [Online]. Available: https://arxiv.org/abs/1812.04783

[26] F. Feng, J. Wu, W. Sun, Y. Wu, H. Li, and X. Chen, "Haze forecasting via deep LSTM," in *Proc. Asia–Pacific Web (APWeb) Web-Age Inf. Manage. (WAIM) Joint Int. Conf. Web Big Data*, 2018, pp. 349–356.

[27] C.-J. Huang and P.-H. Kuo, "A deep CNN-LSTM model for particulate matter (PM2.5) forecasting in smart cities," *Sensors*, vol. 18, no. 7, p. 2220, 2018.

[28] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks*, vol. 3361, no. 10. Cambridge, MA, USA: MIT Press, 1995.

[29] P.-W. Soh, J.-W. Chang, and J.-W. Huang, "Adaptive deep learning-based air quality prediction model using the most relevant spatial-temporal relations," *IEEE Access*, vol. 6, pp. 38186–38199, 2018.

[30] Z. Pan, Y. Liang, J. Zhang, X. Yi, Y. Yu, and Y. Zheng. (2018). "HyperST-Net: Hypernetworks for spatio-temporal forecasting." [Online]. Available: https://arxiv.org/abs/1809.10889

[31] V. Reddy, P. Yedavalli, S. Mohanty, and U. Nakhat. (2017). *Deep Air: Forecasting Air Pollution in Beijing, China*. [Online]. Available: https://www.ischool.berkeley.edu/sites/default/files/sproject_attachments/deep-air-forecasting_final.pdf

[32] T. C. Bui, V. D. Le, and S. K. Cha. (2018). "A deep learning approach for forecasting air pollution in South Korea using LSTM." [Online]. Available: http://arxiv.org/abs/1804.07891v3

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

[34] H. Wang, B. Zhuang, Y. Chen, N. Li, and D. Wei. (2018). "Deep inferential spatial-temporal network for forecasting air pollution concentrations." [Online]. Available: https://arxiv.org/abs/1809.03964

[35] Y. Liang, S. Ke, J. Zhang, X. Yi, and Y. Zheng, "GeoMAN: Multi-level attention networks for geo-sensory time series prediction," in *Proc. IJCAI*, 2018, pp. 3428–3434.

[36] W. Cheng, Y. Shen, Y. Zhu, and L. Huang, "A neural attention model for urban air quality inference: Learning the weights of monitoring stations," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 1–8.

[37] Z. C. Lipton, J. Berkowitz, and C. Elkan. (2015). "A critical review of recurrent neural networks for sequence learning." [Online]. Available: https://arxiv.org/abs/1506.00019

[38] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1310–1318.

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

[40] R. Dey and F. M. Salemt, "Gate-variants of gated recurrent unit (GRU) neural networks," in *Proc. IEEE 60th Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Aug. 2017, pp. 1597–1600.

[41] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.

[42] K. Cho *et al.* (2014). "Learning phrase representations using RNN encoder-decoder for statistical machine translation." [Online]. Available: https://arxiv.org/abs/1406.1078

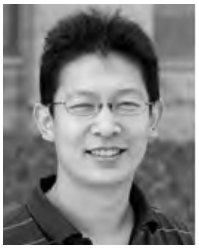
[43] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. (2017). "Convolutional sequence to sequence learning." [Online]. Available: https://arxiv.org/abs/1705.03122

[44] M. Luong, H. Pham, and C. D. Manning. (2015). "Effective approaches to attention-based neural machine translation." [Online]. Available: https://arxiv.org/abs/1508.04025

[45] T. S. Lira, M. A. Barrozo, and A. J. Assis, "Air quality prediction in Uberlândia, Brazil, using linear models and neural networks," *Comput. Aided Chem. Eng.*, vol. 24, pp. 51–56, Jan. 2007.

BO LIU (M'14–SM'17) received the B.S. degree from the Department of Automation, Beijing Institute of Technology, Beijing, China, and the M.S. and Ph.D. degrees from the Department of Automation, System Integration Institute, Tsinghua University, Beijing, in 2003 and 2008, respectively. She was with the NEC Laboratory China, as a Researcher (2008–2010) and (2013–2015). She was a Research Professional with the Computation Institute, The University of Chicago, Chicago, IL, USA, and the Mathematics and Computer Science Division, Argonne National Laboratory, Lemont, IL, USA, from 2011 to 2012. She joined the Beijing University of Technology as an Associate Professor, in 2015. She has authored more than 50 articles and inventions. Her current research interests include big data, data mining, machine learning, cloud computing, scientific workflow, Semantic Web, and ontology reasoning.

SHUO YAN received the B.S. degree from the School of Software Engineering, Beijing University of Technology, Beijing, China, in 2016, where he is currently pursuing the master's degree. His current research interests include big data, data mining, and machine learning.



JIANQIANG LI received the B.S. degree from the Beijing Institute of Technology, 1996, and the Ph.D. degrees from Tsinghua University, in 2004. He was a Researcher with the National University of Ireland, Galway (2004–2005). He was with NEC Labs, China, as a Researcher (2005–2013). He joined the Beijing University of Technology as a Beijing Distinguished Professor, in 2013. His research interests are in Petri nets, data mining, information retrieval, and Big data.

GUANGZHI QU (M'03–SM'10) received the B.E. and M.E. degrees from the Department of Computer Science and Engineering, Beihang University, and the Ph.D. degree in computer engineering from The University of Arizona, in 2005. He joined the Computer Science and Engineering Department, Oakland University, in 2007, where he is currently an Associate Professor. His research areas include data mining, machine learning, operating systems, and program analysis. He was the Conference Co-Chair of the 2014 International Conference on Machine Learning and Applications (ICMLA).

YONG LI received the B.S., M.S., and Ph.D. degrees from the Department of Aircraft Engineering, Beijing Institute of Technology, Beijing, China, and the Department of Communications, Calculations, Chinese Academy of Sciences, Beijing, in 1997 and 2008, respectively. He was with the ZTE laboratory as a Researcher (2000–2005), then in technical division of the China Unicom (2005–2009). He joined the Beijing University of Technology as an Associate Professor, in 2009. He has authored more than 40 articles and inventions. His current research interests include big data, man–machine interaction, neural networks, and machine learning.

JIANLEI LANG received the B.S. and Ph.D. degrees from the Department of Environment Science and Technology, College of Environmental and Energy Engineering, Beijing University of Technology, Beijing, in 2008 and 2013, respectively. He was with the Beijing University of Technology as a Lecturer (2013–2017), and has been an Associate Professor, since 2017. He has published or authored more than 50 articles and patents. His current research interests include environmental science, interdisciplinary research of environmental science and mathematics, data mining, and chemistry.

RENTAO GU (S'06–M'11) received the B.E. and Ph.D. degrees from the Beijing University of Posts and Telecommunications, Beijing, in 2005 and 2010, respectively. From 2008 to 2009, he was a Visiting Scholar with the Georgia Institute of Technology, Atlanta, GA, USA. He joined the Beijing University of Posts and Telecommunications (BUPT), in 2010, and is currently an Associate Professor with the School of Information and Telecommunication Engineering, BUPT. His current research interests include intelligent information processing and communication networks.