IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON URBAN COMPUTING & WELL-BEING IN SMART CITIES: SERVICES, APPLICATIONS, POLICYMAKING CONSIDERATIONS

Received June 27, 2019, accepted July 22, 2019, date of publication August 1, 2019, date of current version January 16, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2932413

Reinforcement Learning for Improving the Accuracy of PM2.5 Pollution Forecast Under the Neural Network Framework

SHUO-WEN CHANG[1], CHUNG-LING CHANG[2], LONG-TIN LI[3], AND SHIH-WEI LIAO[2]

[1]Department of Electrical Engineering, National Taiwan University, Taipei 10617, Taiwan
[2]Department of Computer Science, National Taiwan University, Taipei 10617, Taiwan
[3]Department of Business Administration, National Taiwan University, Taipei 10617, Taiwan

Corresponding author: Shuo-Wen Chang (r06921008@ntu.edu.tw)

ABSTRACT Air pollution seriously damages human health on a large scale. Although earlier works have improved a variety of predictive models of air pollution, the ability to accurately predict air pollution indices remains elusive. Time series prediction plays an important role in many fields. Some predecessors have experimented with artificial neural networks (NNs), combining linear autoregressive integrated moving average (ARIMA) models with nonlinear NN models. The typical assumption is that time series has a long signal with no white noise. However, a real-time short signal with white noise is common. The methods in the literature also do not guarantee that the prediction error of an NN model is minimized. Therefore, we propose the use of reinforcement learning (RL) to predict future PM2.5 values. First, we use the *Q*-learning algorithm in RL based on its state characteristics on the NN model. Second, we select the input with different input dimensions and values of time delay, calculate the best strategy, and evaluate the computational complexity of our RL algorithm. Finally, we show that we effectively reduce the prediction error of the NN models.

INDEX TERMS Smart city, smart environments, urban computing, autoregressive integrated moving average, time series, neural network, reinforcement learning, Q-learning.

I. INTRODUCTION

Air pollution has been a public health issue on an international scale. The most dangerous one is a particulate matter with an aerodynamic diameter of less than 2.5 mm, called PM2.5. PM2.5 not only destroys the ecological environment but also poses a great threat to human health. It increases the incidence of respiratory and heart disease [1]. PM2.5 is more likely to cause serious respiratory illness that leads to possible termination of life. Many institutions and organizations such as the World Health Organization (WHO) have shown that at least 2 million people die of air pollution each year.

Although various models and architectures [2]–[4] regarding the PM2.5 predictions have been presented, the ability to accurately predict the PM2.5 concentration index is still limited. In our study, the PM2.5 univariate time series is used as an example. The Autoregressive Integrated Moving Average (ARIMA) model is used for analysis. ARIMA is a well-known model used for air pollution simulation and analysis. Predicting time series data with high accuracy for linear components can effectively predict univariate time series. Nevertheless, the ARIMA model cannot capture nonlinear characteristics well. In comparison, our research combines ARIMA and the Neural Network (NN) model to deal with the nonlinear characteristics.

Time-series prediction plays an important role in many areas [5]. It has been dominated by linear models such as Autoregressive (AR) and Moving average (MA) [6]. However, these models are difficult to deal with nonlinear and time varying data. In order to overcome this difficulty, predecessors have tried many artificial neural networks [7]. While using the neural network model for time-series prediction, NN models perform future value prediction [8] based on past historical data via diffusion models and ARIMA. Two research groups, Zhang [10] and Kugiumtzis [11] show that the neural network models [9] may enhance the time-series analysis. Overall, other researches in this field [7], [10], [11] aims to boost the nonlinear characteristics of the ARIMA model.

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

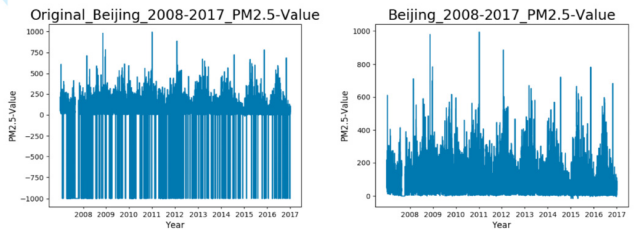

FIGURE 1. 2008-2017 historical data (left), after the first-order linear interpolation of the data (right).

Many of these methods assumed the input data set contains long signal and no white noise data. In real world, many real time series data has short signal with white noise. Those methods can not guarantee that the prediction error of the neural network model is minimized. The focus of our work is to use its characteristics to select the input dimension of the neural network model and the time delay between inputs. The input dimension is the input number of past values for network to predict the future values. The time delay is the delay interval between adjacent inputs. These two parameters determine the structural computational complexity and accuracy of the neural network model in time-series prediction. With appropriate selections of these two parameters, the neural network model can be simplified into a simple structure. As a result, the model is more accurate and of great significance.

In the past, PM2.5 was predicted using supervised learning [12]–[14]. In comparison, we propose a method based on reinforcement learning to predict the future PM2.5 data value. We use the Q-learning algorithm [15] to automatically calculate three functions: First, finding the best or near-optimal pairing of the input dimension and time delay of the neural network model. Second, evaluating the Q value of the input dimension. Finally, acquiring time delay of the derived pair during the Q-learning process. The root-mean-square error (RMSE) is used as our prediction error. So, we propose a whole new method for the structure of PM 2.5 prediction.

For our training dataset, we utilize the data published by the US Department of State's Mission for China Air Quality Monitoring Program. It includes hourly data for the following five regions and several attributes. Only Beijing, China, from 2008 to 2017, as Fig. 1, has been used for nearly 10 years of historical data on air quality as a model training set.

For the validation dataset, we used the 40-day historical data published on Taiwan LASS Environmental Sensing Network System. The validation of data from a different region enables our model to predict PM2.5 value.

II. BACKGROUND AND RELATED WORK

Unlike current methods [16]–[18] using neural network model, support vector machines and regression model to predict PM 2.5. There are plenty of models using ARIMA to perform air pollution prediction [19]–[22].

Andonis Papaleonidas et al. proposed a hybrid model for real time air pollution monitoring. They used Reinforcement learning to enhance the learning capability of their model based on Artificial Neural Networks (ANN) and Fuzzy Rule [23]. Walraven et al. used reinforcement learning [24] to optimize the traffic flow problem which can also be formulated as a Markov Decision Process (MDP). They used Q-learning to learn policies dictating the maximum driving speed that is allowed on a highway.

So far, the models combine ANN with ARIMA [20], [22] to perform air quality forecasting. Also, the model integrates ANN with Reinforcement learning to monitor air quality [23]. Nevertheless, the model which can combine the advantages of reinforcement learning, ARIMA, and neural networks as sub-systems have yet to be proposed.

A. AUTOREGRESSIVE INTEGRATED MOVING AVERAGE MODEL (ARIMA)

The goal of ARIMA is to convert the sequence data into a constant sequence after the non-constant sequence is processed in a differential manner and then predict it through the model of the Autoregressive Moving Average model. For example, the random walk process does not have a fixed average sequence. Then, it becomes a sequence composed of smoothness and make predctions through the Autoregressive Moving Average model.

The extended ARIMA(p, d, q) model of the ARMA(p, q) model can be expressed as:

$$\left(1-\sum_{i=1}^{p}\phi L^{i}\right)\left(1-L^{d}X_{t}\right)=\left(1+\sum_{i=1}^{q}\theta_{i}L^{i}\right)\varepsilon_{t} \qquad (1)$$

The p in Autoregressive represents the Autoregressive term, q in Moving Average represents the Moving Average, and d is the number of differential orders. Differential order is needed to make the time-series a stationary sequence and is greater than zero.

X_t represents the current forecast value and L stands for the lag operator, meaning that it is equivalent to a time pointer, and the sequence value is multiplied by the lag operator. For example, $L^{k}X_{t}=X_{t-k}$. ARIMA model construction process:

1) First, the time-series data is used for stationary identification using the features of the auto-correlation function and the partial autocorrelation function.
2) Then, if the recognition result time-series data is non-stationary, differential processing is taken until the time series data is identified as stationary.
3) Finally, the model is judged by the recognition result, and the characteristics are presented according to the autocorrelation function (ACF) and the partial autocorrelation function (PACF).

Autocorrelation is in the ordered random variable time-series. Compared with the variable sequence itself, autocorrelation shows the correlation of the same sequence at different timings. The partial correlation function is the relationship between the time-series and a short time-series observation after the partial interference is removed.

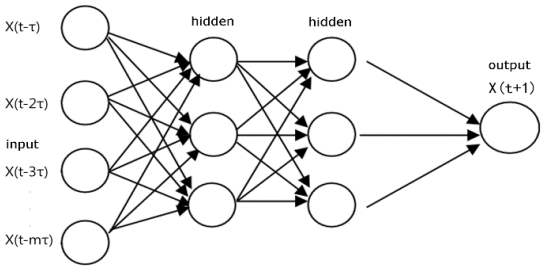

FIGURE 2. Appropriate selection of m and τ to build time series predictions for the NN model.

B. NEURAL NETWORK MODEL (NN)

A feedforward neural network is a type of multi-layer perceptrons that contains multiple neurons in each layer. The neurons in the same layer are disconnected from each other, and the transmission of information between layers is performed in only one direction. The first layer is called the input layer and the last layer is the output layer. The middle is the hidden layer, and the hidden layer can be multi-layer [16]. And the last layer is the output layer.

Time-series prediction plays an important role in many fields. It has been dominated by linear models such as Autoregressive (AR) and Moving Average (MA). However, these models are difficult to deal with nonlinear and time-varying data. In order to overcome this difficulty, predecessors have used the NN model for time-series prediction and predict future values based on past history of data instances; the emphasis is on the input dimension and the time delay between inputs. The input dimension is the number of past values that enter into the network to predict future values. The time delay is the delay interval between adjacent inputs. These two parameters determine the structural computational complexity and accuracy of the NN model in time series prediction.

Consider a scalar time-series such that the delay coordinates derived from the time-series are represented as follows:

$$x(t) = (x(t-\tau), x(t-2\tau), \ldots, x(t-m\tau))^T \qquad (2)$$

In NN modeling, $x(t)$ represents the t-th data in the time-series, m represents the input dimension of the phase space, τ represents the time delay, and m represents the number of input nodes of the Neural Network. The pairing of m and τ determines the structure of the NN model as well as its computational complexity and prediction accuracy. In the design of NN models, the number of hidden nodes is also an important issue.

In our work, the main focus is finding the appropriate choice of m and τ for the NN model for constructing time-series predictions. The time-series prediction using the NN model is shown in the Fig. 2.

In a hybrid model time-series analysis, we assume that the data instance is long signal and no white noise, and then determine m and τ values for time-series data.

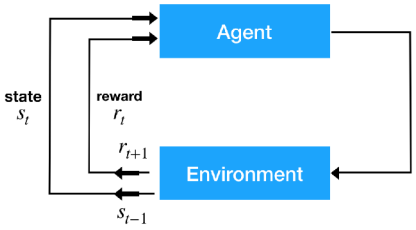

FIGURE 3. Reinforcement Learning model and decision-making process.

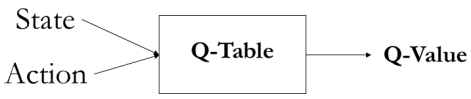

FIGURE 4. Q- Learning process. Q-table is a table that stores Q values with state and action as indexes.

C. REINFORCEMENT LEARNING

Reinforcement learning uses self-adjustment and optimization solutions from environmental feedback. This can be understood as automated systems based on agents, goal-oriented learning, and decision making. The environment is everything that the agent interacts with and it is not fully controlled by the agent. In each calculation step t, the environment representation received by the agent is called the state, represented by $s_t \in S$, where S is the set of all possible states.

In response to the state st, the policy selection action is denoted as π_t in the $a_t \in A(s_t)$ according to the current policy. $A(s_t)$ is one of the behaviors available in state s. At time $t+1$, the agent receives a numerical reward expressed as $r_{t+1} \in R$ from the environment as a result of applying in the state s_t. Afterwards, a new state s_{t+1} will be emitted back to the agent. Set the variable strategy as π, the expected return of the state s is called the value function, expressed as $V(s)$. The expected return from the state s after taking action a is the action value function, or the Q-value, which is expressed as $Q^{\pi}(s, a)$. The end goal of reinforcement learning is to find the best decision from the learning process and maximize the long-term return of automated agent-based systems.

Reinforcement learning can also be considered as a task-independent learning algorithm. It applies to issues that have no supervisory information, but only feedback from the external environment. The NN model is used to determine the policy to pick m and τ for time-series prediction. In addition, reinforcement learning selects the input dimension and time delay in a self-adjusting optimization manner. It also has feedback from autonomous and environmental data and can well overcome the shortcomings of existing methods highlighted by the neural network. In recent years, many reinforcement learning algorithms have been proposed, one of which is Q-learning, which attempts to approximate the optimal action-value function. In our study, the Q-learning algorithm was used to selecting strategies as well as updating neural network input dimensions and time delays.

Q-learning process is shown in Fig. 4. The first step is to create a Q-Table that is used to estimate the best outcome Q value for a particular action in a particular state.

Q-Learning process. The Q-table is a table that stores Q values with state and action as indexes. However, this table can only be created if the state and action are limited. For example, in the our model, the state has only six attribute each being a limited value. Each row in the Q-Table represents a state, and each column represents an action. Q-learning can find the best choice strategy in the Markov decision process (MDP) by learning the action-value function.

The state of the real situation may be infinite, and such a Q-Table will be infinite. The solution to this problem is to realize the input state of the Q-Table through the neural network and output the q values of different actions.

Q-learning is expressed mathematically as follows:

$$Q^{new}(s_t, a_t) = Q(s_t, a_t) + a\{r_{t+1} + \gamma_a^{max} Q(s_{t+1}, a) - Q(s_t, a_t)\} \quad (3)$$

The Q-Table update begins with the max a $Q(s_{t+1},a)$ part of Equation 6, and assuming $t = 1$, the value of $Q(s_2, a_1)$ is larger than $Q(s_2,a_2)$. Since recent action matters more than a long ago action, $Q(s_2, a_1)$ is multiplied by an attenuation value gamma, assumed to be 0.9, plus the reward R obtained when reaching s1. We use it as the real $Q(s_1, a_1)$ which has a value of 17. Our study then estimate the value of $Q(s_1, a_1)$ based on Q-Table, and we used the difference between the actual value 17 and the estimated value of 8 to multiply by the learning efficiency. After alpha, the hypothesis is 0.6, and the value of the estimated $Q(s_1,a_1)$ is increased to become the new estimate of 13.4. This is how Q-learning makes decisions and learns to optimize decisions in the rewards section.

III. ARCHITECTURE DESIGN

The purpose of using reinforcement learning is to determine the action strategy more accurately by selecting the input dimension and time delay. The determined action strategy can reach the optimal or near-optimal predictions, and is used in the NN model of time-series prediction. The unique mechanism of action strategy focuses on selecting the input dimension [17], [18] and time delay while maintaining a high level of prediction accuracy. The choice of these two parameters is mapped to the important task of intensive learning task. The architecture of our proposed system is shown in Fig. 5.

Considering the NN model and time-series data as the external environment, the actions taken by the agent in response to the external environment state, and then defined as the input dimension and time delay of the NN model via the Q-Table comparison in reinforcement learning, i.e. $a = (m, \tau)$, where m represents the input dimension and τ represents the time delay. Time delay can be selected as a setting of parameter, as Fig 6.

Assuming that the state of time $t+1$ is to be predicted, our study selects the input parameter dimension by m. Assuming $m = 3$ here, the input of the NN model has only three quantities. Parameter time delay τ of the input can also be selected. Assuming $\tau = 2$, each time the input parameter is selected, data before $t-2$ will be selected as the input.

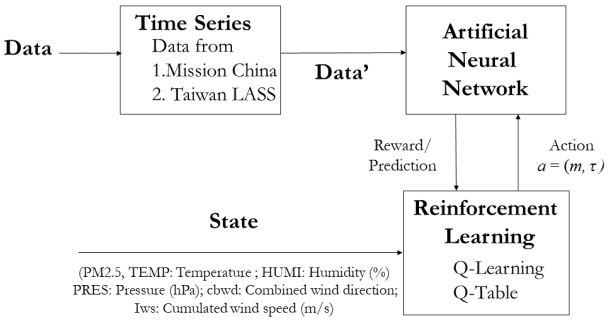

FIGURE 5. RL-ARIMA-NN model architecture. The model is based on reinforcement learning module which uses the feedback of autonomous environmental data to select the input dimension and time delay for Feedforward Neural Network Module and Time-Series Module.

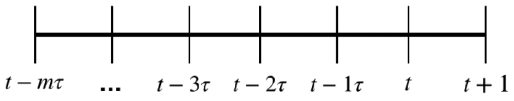

FIGURE 6. The parameter of time delay for the input by τ time unit.

The RL-ARIMA-NN model architecture is composed of three modules: Reinforcement Learning Module, Feedforward Neural Network Module, and Time-Series Module. Firstly, The Reinforcement Learning module is initialized by a set of initial states and actions. Next, the model learns and then updates other states and actions during the model learning process. The Time-Series module will organize the data in a smooth time-series, and then analyze it by ARIMA model to determine whether to adopt Autoregressive model, Moving Average model or Autoregressive Moving Average model strategy, then the time-series Data of ARIMA model's prediction is passed to the NN model for learning. The Feedforward Neural Network Module (NN model, As shown in Fig. 8) uses actions made by the Reinforcement Learning module and the data from the Time Series module to train itself as well as update the weights by minimizing the mean square error. The prediction error is expressed as the root-mean-square error (RMSE), and the calculated reward values are passed onto the reinforcement learning module. Then the update of Q-Table will be executed so that the model can predict more accurately and effectively next time. As shown in Fig. 7, Reinforcement Learning module learns and updates Q-Tables based on feature status and rewards.

A. REINFORCEMENT LEARNING MODULE

The first step of the module is to create a Q-Table that is used to estimate the best resulting Q value for a particular action in a particular state. Each row in the Q-Table represents a state and each column represents an action. Q-learning can find the best strategy in the Markov decision process (MDP) by learning the action-value function.

In this study, Q-Table takes S_t as the input state at time t, and S_t is the stationary time-series data processed by time-series. In the proposed RL-NN method, the attribute values

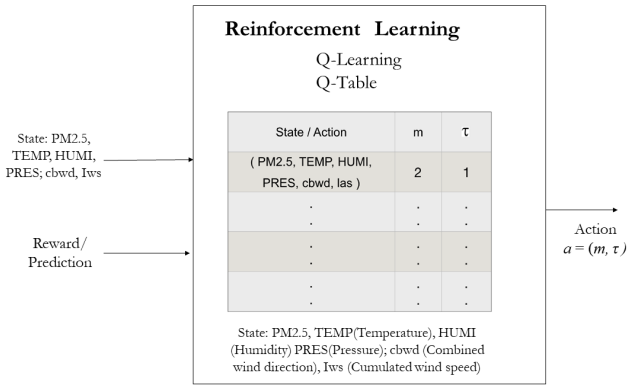

FIGURE 7. The reinforcement Learning module. The model learns and updates Q-Tables based on feature status and rewards.

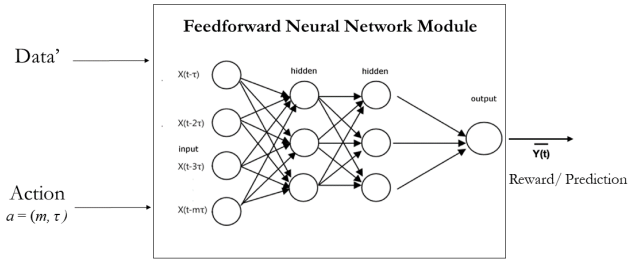

FIGURE 8. The Feedforward Neural Network module.

are used as the state variables for learning. It contains six attributes as the state parameters PM2.5: PM2.5, TEMP: temperature, HUMI: humidity, PRES: atmospheric pressure, cbwd: integrated wind direction and las: cumulative wind speed. These attributes are the state variables, which well describes each state in the Reinforcement learning, normally denoted as S. For more detail of how the learning process works, please reference the theory of Reinforcement Learning [27].

We define initial action $a = (m,\tau)$ at $t = 0$. And then, s_t and $a = (m, \tau)$ at $t = t_i$ will automatically learn and update m and τ *value* during the training procedure of the RL-ARIMA-NN model. The architecture of Reinforcement Learning module is shown in Fig. 7.

RL-ARIMA-NN model is started in the second step, It learns and updates the Q-Table after selecting the next action and reward based on the current status and Q-Table. Feedforward Neural Network module, as shown in Fig. 8, adjusts the weights of the NN model and makes predictions with selected m and τ.

B. FEEDFORWARD NEURAL NETWORK MODULE

Initially, when s_t is fed as a parameter into the Q-Table, the at action is taken, and the at action contains $a = (m,\tau)$ as a parameter of the neural network model, which then generates a prediction result $\overline{y(t)}$, the reward for the action a_t is defined as the neural network model prediction result $\overline{y(t)}$, which can evaluate the prediction result of the RL-ARIMA-NN model accuracy and error. The predicted result $\overline{y(t)}$ is compared with the stationary time-series data value provided by the actual

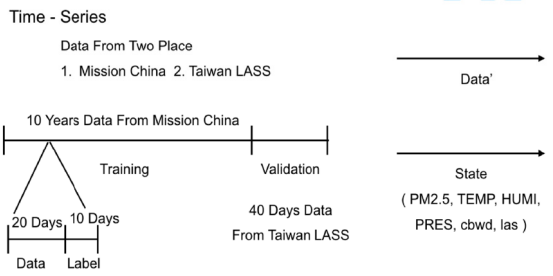

FIGURE 9. The correlation between time-Series module, output data and status of features.

time-series, which is called the actual value $y(t)$, and the mean square accuracy is calculated. The prediction error is expressed as the root-mean-square error (RMSE), defined as follows:

$$RMSE(y(t), \overline{y(t)}) = \sqrt{\frac{\left(\sum_{t=0}^{n}(y(t) - \overline{y(t)})^2\right)}{n}} \tag{4}$$

where $\overline{y(t)}$ is the actual value while $y(t)$ is the predicted output of the predicted RL-ARIMA-NN model. When the time t is calculated by RMSE, the reward for st taking the at action is defined as:

$$r = (1 - RMSE(y(t), \overline{y(t)}))^2 \tag{5}$$

The design of (5) here indicates that higher prediction error gives less reward for the action, and vice versa. Since the range of actual values is a contiguous space, discretization is performed to derive various states. This is how Q-learning makes decisions and learns to optimize decisions in the reward system.

C. TIME-SERIES MODULE

Time-Series module outputs data and feature status are shown in Fig. 9.

Our work leverages a training set and validating set to train the RL-ARIMA-NN model. We take the historical data published in the US Department of State's Mission China Air Quality Monitoring Program, which includes Beijing's historical data on air quality for nearly 10 years from 2008 to 2017, as a training set. The 40-day historical data published by the Taiwan LASS Environmental Sensing Network System is used as a verification set. We verified on the data from different region to prove the effectiveness of our model in the case of providing the same attribute state parameters in different regions.

Since the data used in the time-series analysis is continuous, the first-order linear interpolation method is used to estimate the data of the missing observations, and the data is collated. Then the duration of the total observation data of Mission China air quality is 10 years. 3195 data points are hashed and the entire data set is divided into two parts for fragmentation learning. Each time the clip learning takes 30 days of data, the first 20 days of data is treated as a prediction label, and the data of the next 10 days is used as a

ground truth label. To use the difference between the output from the prediction and the correct value, the neural network calculates the loss, computes the gradients to update weights.

First, the time-series data is used for stationary identification using the features of the autocorrelation function and the partial autocorrelation function. If the recognition result of the time-series data are non-stationary, differential processing will be taken, until the time-series data is identified as stationary. The training data is a non-constant sequence, which needs to be processed differentially to become a constant sequence.

According to the smoothing process of the corresponding time-series model established after the recognition, an Autoregressive model is established when the partial autocorrelation function is truncated and the autocorrelation function is smeared. A Moving Average model is established when the partial autocorrelation function is smeared and the autocorrelation function is truncated. If both the partial autocorrelation function and the autocorrelation function are smeared, the sequence is suitable for the Autoregressive Moving Average model.

Then the time-series data of the Autoregressive Moving Average prediction is transmitted to the neural network model. The neural network model is trained to adjust and update the weights as well as the Q-Table, so that the model can predict the future more accurately and effectively.

The 40-day historical data published by the LASS environmental sensing network system in Taiwan is also used as a first-order linear interpolation method to estimate the data of the missing observations, and the data is collated, and then put into the trained RL-ARIMA-NN model for verification. Because there is already a well-trained Q-Table at this time, knowing that the m and τ parameters need to be taken as the inputs in different states. Data collected by another region is taken as verification to show our model has the ability to apply on different regions.

IV. RESULTS AND DISCUSSION

Air pollution prediction is certainly the solution for the worse and worse air quality. Although multiple types of prediction models and architectures have been improved, the ability to accurately predict is still limited. Actual time-series data contains a combination of linear and nonlinear features, but the use of ARIMA model or NN models cannot adequately describe the characteristics of actual data.

The purpose of this study uses Reinforcement learning to be more precise in determining the input dimension and time delay. In conclusion, the mixture of the three models would have a better predictive effect than a single model. The linear part of our architecture is modeled by the ARIMA model. The predicted result is overline Lt. The original sequence and overline Lt have a residual of et. The formula is as follows:

$$e_t = y_t - \overline{L}_t \tag{6}$$

The residual et implies the nonlinear relationship in the original sequence. Therefore, our study uses the NN model to approximate this nonlinear relationship and assumes that there are at least n inputs in the NN model. The formula is described as follows:

$$e_t = f\left(e_{t-1}, e_{t-2}, \ldots, e_{t-n}\right) + \varepsilon_t \tag{7}$$

In this equation, f is a nonlinear function determined by NN, and t is a random error. The predicted value is et estimated by NN, and the predicted value is recorded as N_t. Then the results predicted by the hybrid model can be written as:

$$\overline{y_t} = \overline{L_t} + \overline{N_t} \tag{8}$$

The hybrid model is characterized by an ARIMA model for linear partial prediction and a NN model for non-linear partial prediction. We used comprehensive application of ARIMA and NN models to fully achieve their respective strengths and to achieve the purpose of improving the prediction effect.

In our study, the dual model serial superposition method, ARIMA-NN, is improved through reinforcement learning of the Q-learning, which makes the NN model take a better parameters of input dimension and time delay. The first model is built with ARIMA model by dividing 20,000 data points into two subsets. The initial 16000 data points, are used for training data, and the remaining 4000 data points are used for testing. The Root Mean Square Error (RMSE) is used to evaluate the predictive performance on the test set. The residual of the ARIMA model is assigned to the neural network to establish a second model. Eventually, the predicted values of the two models are superimposed into the final predicted value. In addition, the 40-day historical data published by the Taiwan LASS Environmental Sensing Network System was used as a validation set. For example, Zhang uses this method to solve classic examples of three-time series: sunspot problem(1700−1987,288yeardata), Canadian lynx problem (1821−1934, 114 data per year), Q-Table. With this method, we have trained and verified the RL-ARIMA-NN model by comparing with the accuracy of ARIMA-NN and other significant CNN models.

A. DATA

Since the data used needs to be continuous in the time-series analysis, we collated the data before using the observation data to train the model of our study. Therefore, our study uses the first-order linear interpolation method to estimate the data of the missing observations. Therefore, the 10-year data is aggregated into the total observations. From 2008 to 2017, a whole dataset of 20,000 data points has been hashed. The entire dataset is then used as a fragment for the model to learn. We captured 30 days of data each time; the first 20 days of data are regarded as a predict label and the next 10 days of data are considered as the ground truth labels.

The 40-day historical data disclosed by the Taiwan LASS environmental sensing network system adopts the best strategy according to the state numbering parameter and the data is modeled directly to the ARIMA model. After determining several possible tentative modes, AIC is used as the criterion for judgment, and a model with a smaller AIC is selected, as shown in Fig. 11. The AIC values of the tentative modes

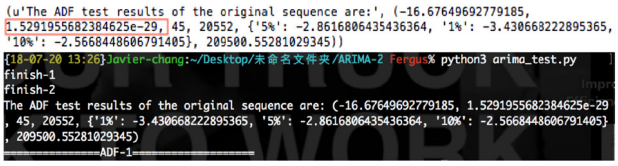

FIGURE 10. ADF Augmented Dickey-Fuller test.

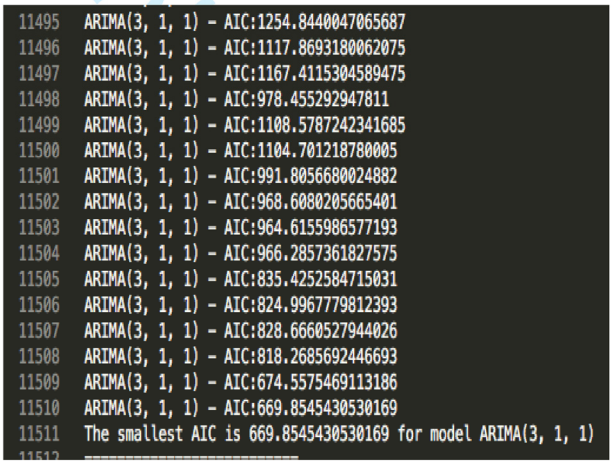

FIGURE 11. We found the smallest AIC values of ARIMA(3, 1 ,1) through our method.

TABLE 1. ARIMA tentative model AIC values.

pdq	AIC	pdq	AIC
003	710.24448	012	690.843490
103	710.430246	102	685.990614
013	704.25332	300	736.909788
113	696.38771	301	725.374845
210	698.46934	310	729.545354
201	700.32840	311	669.854543

are compared in Table I, and the AIC values are smaller when the modes of p=3, d=1, and q=1 are obtained.

B. ARIMA MODEL CONSTRUCTION

The PM2.5 data per hour from 2008 to 2017 is shown in Fig. 1. Box and Jenkins proposed three model evaluation methods: the parameter estimation, the pattern identification, and the AutoCorrelation Function (ACF).

According to the Auto-Correlation Function test, the absolute value of the Auto-Correlation Function is kept smaller and smaller for a long time, and it is judged as a smooth sequence. The model judges that before performing a time-series analysis. It is necessary to judge the model to determine whether the sequence is stable. If the sequence is unstable, we perform the difference processing until it is smooth enough. Our study used an expansion of the Dickey-Fuller (ADF) single root assay. The ADF detection feature describes that the appropriate factor variation is added to the regression equation as shown in Fig. 10. Finally, the hypothesis error term in the DF test is corrected to have no self-correlation,

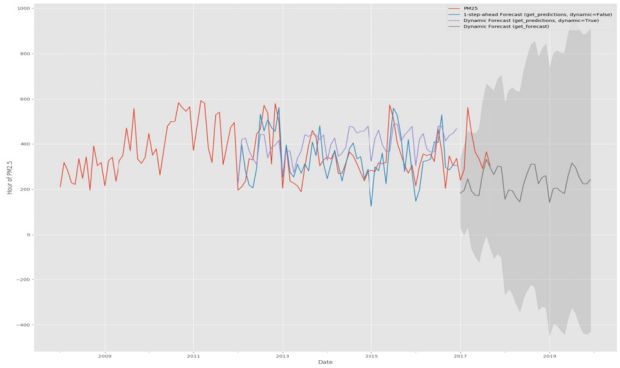

FIGURE 12. ARIMA model (3,1,1) prediction map.

and the white noise problem caused by the Moving Average term is ignored.

As a result of the ADF test, the p value of the return value is significantly less than 0.05 (p value = 1.5291 955682384625e-29), and therefore, the time series is judged to be a smooth sequence. The model is diagnosed and the and the British pound dollar exchange rate issue (1980−1993, 731 weekly data)

The results predicted using the constructed model (3, 1, 1) are shown in Fig. 12. Red line: historical actual value. Blue line: 1-step-ahead Forecast (get predictions, dynamic = False). Purple line: Dynamic Forecast (get predictions, dynamic = True). Gray line: Dynamic Forecast (get forecast). If it is a static prediction, the actual value will be used for the next prediction.

Our study uses an ARIMA model to perform time-series prediction of PM 2.5. In the case of dynamic prediction, the next predicted value will be used for the following prediction values. From the results, we see that the prediction after time p (2015) starts to become inaccurate and the error increases gradually. Our study uses an ARIMA model to assume that the PM 2.5 value is linear, but the determinants depend on its previous p-value. This can be a relatively good approximation to p time points.

However, over time, the accuracy of the ARIMA predicts PM 2.5 values begins to decline. It is resulted of the non-linearity of the PM 2.5-time series data. Therefore, in order to enhance the prediction accuracy, we make use of other models that can better describe non-linearity.

Then, our study used the appropriate data set to generate the ARIMA (3,1,1) model to predict the PM2.5 concentrations after 5 days. The error results in calculating the Predicted error and RMSE are shown in table 2.

The Fig. 13 shows the error between the actual value and the predicted value of the ARIMA (3,1,1) mode. From the residuals from 2008 to 2017, the residual is assigned to the neural network to establish a second model.

C. RL-NN MODEL CONSTRUCTION

At the end of our study, the difference between the predicted value of the ARIMA model and the actual value is called the

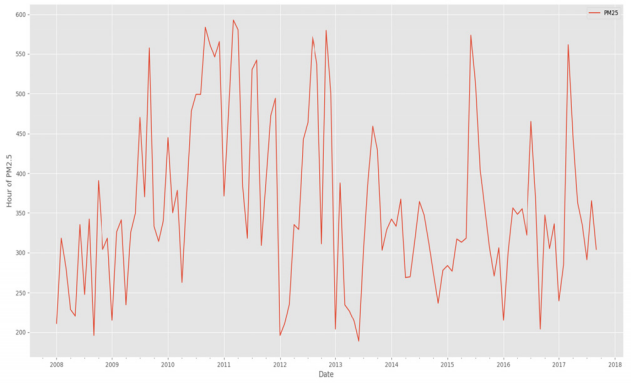

FIGURE 13. The error between ARIMA model (3,1,1) prediction.

TABLE 2. ARIMA tentative model AIC values.

Methods	AIRMA
Predicted error	32.02%
RMSE	84.606%

residual, and the residual is passed to the NN model to make the second model. The RL-ARIMA-NN and the traditional ARIMA-NN model are tested with RMSE. The Predicted Error and the validation data set is then predicted using the best strategies of the two models.

As mentioned previously, the residual 20000 data points are divided into two subsets. The first 18,000 data were used for RL-ARIMA-NN and Neural Network methods as training data, and 30 data sets (batch size =30) were divided into 600 training samples. Finally, the remaining 2000 data points are used for testing, and the root-mean-square error (RMSE) is used to indicate the predictive performance on the test set. In addition, the 40-day historical data published by the Taiwan LASS Environmental Sensing Network System was used as a verification set for verification.

In our study, a three-layer Neural Network is used as the predictive model design in the NN model, and the number of iterations (epochs) is set to 100 times. After each iteration, the NN model adjusts the weight between nodes, and the hidden layer the number of nodes (hidden unit) is set to 128 nodes. The input dimension is set to a maximum of 30 with each time capturing 30 days of data.

Among them, the ARIMA-NN model manually adjusts the parameters with input dimensions (i.e the window size) to find the best combination of strategies. RL-ARIMA-NN uses Q-learning in Reinforcement learning with six attributes as the state parameters: PM2.5, TEMP (temperature), HUMI (humidity), PRES (atmospheric pressure), cbwd (integrated wind direction) and las (cumulative wind speed).

In order to find the best combination of strategies, we choose which input dimension (m) to take in conjunction with the time delay (τ). Nevertheless, it may lead to potential over-fitting problems. In practice, this study attempts to use

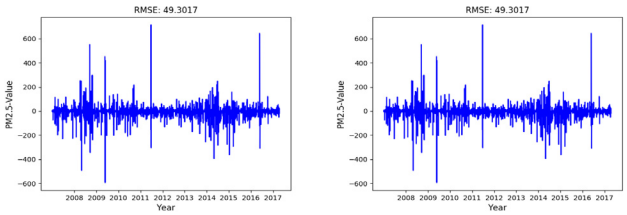

FIGURE 14. The ARIMA-NN model has a short-term prediction input dimension of 6.

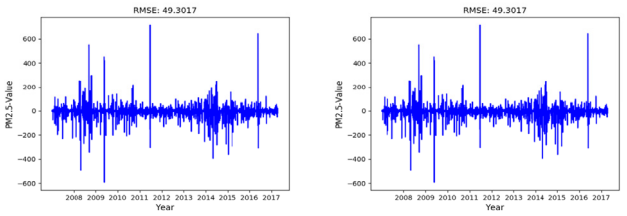

FIGURE 15. The ARIMA-NN model has a short-term prediction input dimension of 5.

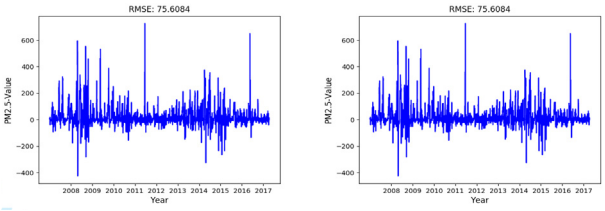

FIGURE 16. The ARIMA-NN model has a long-term prediction input dimension of 18.

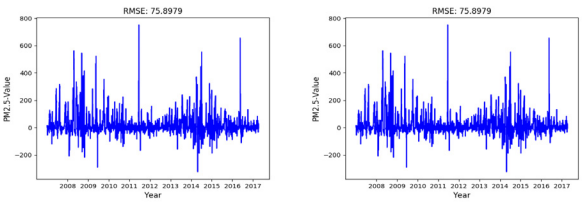

FIGURE 17. The ARIMA-NN model has a long-term prediction input dimension of 17.

the different parameters and structures to obtain the best model in order to balance the variance and bias.

The results of the ARIMA-NN model train on 18,000 data points and are tested at 2000 data points using different input dimensions as shown in Figure 14, Figure 15, Figure 16 and Figure 17. The RMSE and Predicted Error are listed in the Table 3. The table shows the analysis with short-term and long-term. The RMSE and Predicted Error are both lower in the short-term forecast. Because of the time delay, the NN model is preset to a time unit delay for the learning of the time-series. Therefore, the short-term forecasts are more accurate than the long-term prediction. The performance of the model with six input dimensions is slightly better than the model with five input dimensions.

In the long-term prediction part of Table. 3, the performance (both error and RMSE) of model with 18 input dimensions is better than the model with 17 input dimensions. Since it's impossible to change the input state and time delay of ARIMA-NN model, and the overall long-term prediction of

TABLE 3. ARIMA-NN model predicted error and RMSE.

Dim	Predicted Error	RMSE
Forecast 3 days later (short term)		
6	12.60%	49.95
5	14.08%	54.07
4	17.65%	58.10
Forecast after 1 month (long term)		
18	26.47%	75.60
17	28.42%	75.89
16	31.31%	76.48

TABLE 4. RL-ARIMA-NN model predicted error and RMSE.

Dim	Delay	Predicted Error	RMSE
Forecast 3 days later (short term)			
7	1	11.68%	49.95
6	1	12.28%	54.07
5	1	14.65%	58.10
Forecast after 1 month (long term)			
18	9	10.84%	75.60
17	4	17.02%	75.89
16	1	20.82%	76.48

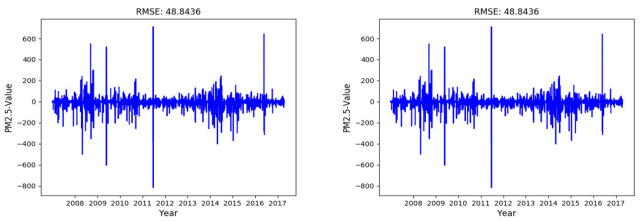

FIGURE 18. The RL-ARIMA-NN model has a short-term prediction with the input dimension 7 and time delay of 1 unit.

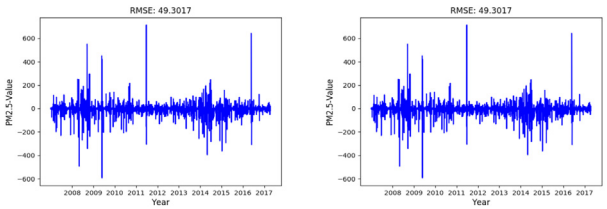

FIGURE 19. The RL-ARIMA-NN model has a short-term prediction with the input dimension 6 and a time delay of 1 unit.

ARIMA-NN model is worse than RL-ARIMA-NN model (which is shown in Table 4) in general.

From the Table 3, the optimal short-term prediction uses the parameter of 6 input dimensions and an optimal input dimension of 8 in the long-term prediction. This may occur because the number of input dimensions' increase enough. The various setting of parameter can be considered as a simulation of a dynamic air pollution system in real world.

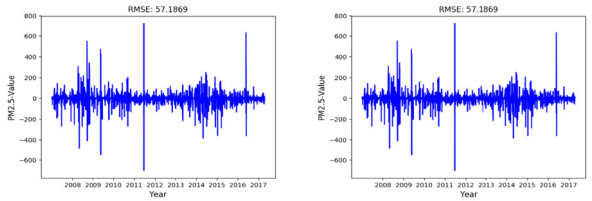

FIGURE 20. The RL-ARIMA-NN model has a long-term prediction with the input dimension 4 and a time delay of 9 units.

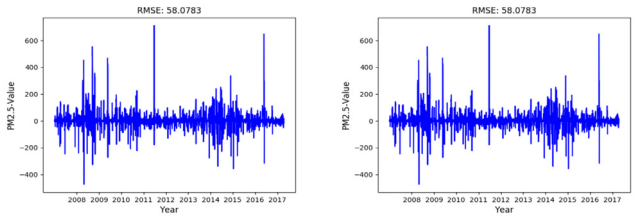

FIGURE 21. The RL-ARIMA-NN model has a long-term prediction with the input dimension 4 and a time delay of 4 units.

In Fig. 18, 19, 20 and 21, we listed other results using different input dimensions and time delay to more easily analyze the RL-ARIMA-NN model. The RMSE and prediction errors are also listed in the Table 4. According to our inference, due to the time delay, the RL-NN model is chosen to be used to learn the time delay of the time series.

Since the combination of input state with the time delay can be optimized automatically with reinforcement learning mechanism, the overall prediction is better than the ARIMA-NN model. We discovered one thing that is worthy to discuss: in the short-term prediction, optimal input dimension is 7 while input dimension of long-term prediction is 4 with time delay of 9 units (there are more choices for the input dimension).

Observing its prediction map of the test data, it is concluded that this model better describes the nonlinear system in the real world and compensates for the accuracy of the ARIMA-NN model's prediction. The combination of the RL and NN models help capture the trend of raw data perfectly.

In Table 5, we listed the results of the training and validation error of four models. We further compared our model with the original ARIMA-NN model, the models which using Convolution Neural Network to estimate air pollution [25] and PM2.5 [26]. RL-ARIMA successfully outperform them in both training and validation performance.

Both two errors of RL-ARIMA-NN model are significantly smaller than those of other models.

In short-term prediction, RL-ARIMA-NN model and ARIMANN model can both create the optimal parameter because the NN model's default time delay is one unit. Therefore, the two models with the same time delay have similar performance (RMSE and Predicted Error).

Overall, the optimized RL-ARIMA- NN model can select the optimal parameters such as input dimension, temperature, wind direction, and humidity, etc. With different input states

TABLE 5. Training and validation error comparison.

Methods	AIRMA	RL-ARIMA-NN
Training error	19.5%	**11.26%**
Validation error	27%	**19%**

and then the model will make the decision to obtain a prediction $S+1$.

The original NN model, however, does not have this step, and there is no way to adjust the time delays and input dimensions (the only way is to manually adjust them as hyperparameter). In other words, RL-ARIMANN can adapt to different applications, and it is very easy to let the model learn different states and find the best strategy. Instead of manually adjusting several parameters repeatedly, RL-based models will find the best strategy automatically. It is believed that the increase in the time delay and the number of input dimensions may improve the accuracy of prediction, better stability.

V. CONCLUSION

We propose a new model based on Q-learning and Reinforcement Learning, which is used for time-series prediction. It estimates the input dimension and time delay between neural network model inputs. Our goal is to construct a NN model which maintain the accuracy with a more and more simple structure. Our method compensates for the lack of nonlinearity of the ARIMA model while maintaining high precision and reducing its computational complexity at the same time.

We improve upon those traditional combinations of ARIMA and Neural Network learning algorithm to predict real time-series data. Shorter and noisier time-series data is no longer an absolute obstacle in improving the prediction error of the NN model.

Experimental results demonstrate the effectiveness of our method. Our study finds that the Autoregressive Moving Average model has good performance to predict short-term time-series changes but loses its predictive ability in long-term changes. This is because the ARIMA model assumes that the relationship between the source and the response is linear, while PM 2.5 is not linear in the long-run. While Neural Networks are good at long-term prediction, the number of hidden layers and the number of nodes in each layer of the Neural Network may affect the performance and efficiency of the verification data. This study does not discuss changes in this number.

We created a universal forecasting system that interacts with and learns from the world, and the interaction with the environment is the advantage of Reinforcement learning. Using Reinforcement learning empowers the agents to better understand their surroundings in their experiments and enables them to possibly gain insights into high-level causality. After overcoming the applicability limitations of each method, our model broadens the domain of predictive technology for sensor network.

After all, this work can confidently provide a scientific and methodological basis for the research in the prevention and prediction of urban haze pollution. All of this is an uncharted territory which may seem bizarre to people from this field of work, nevertheless, we have successfully nurtured this ideal into a new frontier for predicting air pollution.

REFERENCES

[1] Y.-F. Xing, Y.-H. Xu, M.-H. Shi, and Y.-X. Lian, "The impact of PM2.5 on the human respiratory system," *J. Thoracic Disease*, vol. 8, no. 1, pp. E69–E74, 2016.

[2] Y. Zheng, X. Yi, M. Li, Z. Shan, E. Chang, and T. Li, "Forecasting fine-grained air quality based on big data," in *Proc. ACM 21th SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 2267–2276.

[3] D. J. Lary, T. Lary, and B. Sattler, "Using machine learning to estimate global $PM_{2.5}$ for environmental health studies," *Environ. Health Insights*, vol. 9, pp. 41–52, May 2015.

[4] S. Mahajan, L.-J. Chen, and T.-C. Tsai, "An empirical study of PM2.5 forecasting using neural network," in *Proc. IEEE SmartWorld, Ubiquitous Intell. Comput., Adv. Trusted Comput., Scalable Comput. Commun., Cloud Big Data Comput., Internet People Smart City Innov.*, Aug. 2017, pp. 1–7.

[5] C. Voyant, M. Muselli, C. Paoli, and M.-L. Nivet, "Numerical weather prediction (NWP) and hybrid ARMA/ANN model to predict global radiation," *Energy*, vol. 39, pp. 341–355, Mar. 2012.

[6] A. D. Syafei, A. Fujiwara, and J. Zhang, "Prediction model of air pollutant levels using linear model with component analysis," *Int. J. Environ. Sci. Develop.*, vol. 6, no. 7, p. 519, 2015

[7] C. Christodoulos, C. Michalakelis, and D. Varoutas, "Forecasting with limited data: Combining ARIMA and diffusion models," *Technol. Forecasting Social Change*, vol. 77, no. 4, pp. 558–565, 2010.

[8] A. S. Weigend and N. A. Gershenfeld, "Time series prediction: Forecasting the future and understanding the past," Boulder, CO, USA: Westview, 1993.

[9] S. Haykin, *Neural Networks: A Comprehensive Foundation*, vol. 2. Upper Saddle River, NJ, USA: Prentice-Hall, 1999.

[10] G. P. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, Jan. 2003.

[11] D. Kugiumtzis, B. Lillekjendlie, and N. Christophersen, "Chaotic time series Part I: Estimation of some invariant properties in state space," *Model., Identificat. Control*, vol. 15, no. 4, pp. 205–224, 1994.

[12] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 279–292, 1992.

[13] L. Fletcher, V. Katkovnik, F. E. Steffens, and A. P. Engelbrecht, "Optimizing the number of hidden nodes of a feedforward artificial neural network," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, vol. 2, May 1998, pp. 1608–1612.

[14] Z. Sheng, L. Hong-Xing, G. Dun-Tang, and D. Si-Dan, "Determining the input dimension of a neural network for nonlinear time series prediction," *Chin. Phys.*, vol. 12, no. 6, pp. 594–598, 2003.

[15] M. B. Kennel and H. D. I. Abarbanel, "False neighbors and false Strands: A reliable minimum embedding dimension algorithm," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 66, no. 2, 2002, Art. no. 026209.

[16] A. Suleiman, M. R. Tight, and A. D. Quinn, "Applying machine learning methods in managing urban concentrations of traffic-related particulate matter (PM_{10} and $PM_{2.5}$)," *Atmos. Pollut. Res.*, vol. 10, no. 1, pp. 134–144, 2019.

[17] W. Yang, M. Deng, F. Xu, and H. Wang, "Prediction of hourly $PM_{2.5}$ using a space-time support vector regression model," *Atmos. Environ.*, vol. 181, pp. 12–19, May 2018.

[18] J. K. Deters, R. Zalakeviciute, M. Gonzalez, and Y. Rybarczyk, "Modeling $PM_{2.5}$ urban pollution using machine learning and selected meteorological parameters," *J. Elect. Comput. Eng.*, vol. 2017, May 2017, Art. no. 5106045.

[19] U. Kumar and V. K. Jain, "ARIMA forecasting of ambient air pollutants (O_3, NO, NO_2 and CO)," *Stochastic Environ. Res. Risk Assessment*, vol. 24, no. 5, pp. 751–760, 2010.
[20] L. A. Díaz-Robles, J. C. Ortega, J. S. Fu, G. D. Reed, J. C. Chow, J. G. Watson, and J. A. Moncada-Herrera, "A hybrid ARIMA and artificial neural networks model to forecast particulate matter in urban areas: The case of Temuco, Chile," *Atmos. Environ.*, vol. 42, no. 35, pp. 8331–8340, 2008.
[21] A. B. Chelani and S. Devotta, "Air quality forecasting using a hybrid autoregressive and nonlinear model," *Atmos. Environ.*, vol. 40, no. 10, pp. 1774–1780, 2006.
[22] A. Samia, N. Kaouther, and T. Abdelwahed, "A hybrid ARIMA and artificial neural networks model to forecast air quality in urban areas: Case of tunisia," *Adv. Mater. Res..*, vol. 518, pp. 2969–2979, May 2012.
[23] A. Papaleonidas and L. Iliadis, "Hybrid and reinforcement multi agent technology for real time air pollution monitoring," in *Proc. IFIP Int. Conf. Artif. Intell. Appl. Innov.* Berlin, Germany: Springer, 2012, pp. 274–284.
[24] E. Walraven, M. T. J. Spaan, and B. Bakker, "Traffic flow optimization: A reinforcement learning approach," *Eng. Appl. Artif. Intell.*, vol. 52, pp. 203–212, Jun. 2016.
[25] C. Zhang, J. Yan, C. Li, X. Rui, L. Liu, and R. Bie, "On estimating air pollution from photos using convolutional neural network," in *Proc. ACM 24th Int. Conf. Multimedia*, 2016, pp. 297–301.
[26] C.-J. Huang and P.-H. Kuo, "A deep CNN-LSTM model for particulate matter ($PM_{2.5}$) forecasting in smart cities," *Sensors*, vol. 18, no. 7, p. 2220, 2018.
[27] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, pp. 237–285, May 1996.

SHUO-WEN CHANG was born in Kaohsiung, Taiwan. He received the B.S. degree in engineering from National Taiwan Normal University, in 2017. He is currently pursuing the M.S. degree in electrical engineering with the National Taiwan University.

In 2018, he did research project—Software development and signal processing for high throughput silicon multi-electrode array systems—at the Life Science Technology Department, Interuniversity Microelectronics Centre, Belgium, as an International Scholar. His research interests include biomedical image processing, and neural network application.

Mr. Chang was a recipient of the Outstanding Student Achievement Award of National Taiwan University, in 2018, and outstanding youth of National Taiwan University, in 2019.

CHUNG-LING CHANG was born in Taipei, Taiwan. He received the M.S. degree from National Taiwan University, in 2018. From 2016 to 2018, he worked as a blockchain developer and security consultant for blockchain companies. His research interests include blockchain, big data, and deep learning. He is also active in network planning for conferences.

LONG-TIN LI was born in Taiwan, Taipei. He received the high-school Diploma degree from the Taipei European School. He is currently studying technological management in business administration with National Taiwan University.

His research interests include stock exchange, blockchain, machine learning, and big data analysis.

Mr. Li was a recipient of the Outstanding Delegate Award of National Taiwan University Model United Nations, in 2018.

SHIH-WEI LIAO was born in Taipei, Taiwan. He received the bachelor's degree from National Taiwan University, in 1991, and the M.S. and Ph.D. degrees from Stanford University, in 1995 and 2000, respectively.

From 2000 to 2007, he was with Intel Corporation, USA, as an Architect. He worked at Silicon Valley for more than 20 years (Stanford, Google, and Intel). He is currently an Associate Professor with the Computer Science Department, National Taiwan University. He is one of the original authors of Android compiler, Virtual machine, and RenderScript engine. He also publishes extensively on multicores, compilers, programming systems, and blocking and affine partitioning theory, which is included in the Dragon Book. He leads a team that releases the Gcoin Blockchain and CPoW consensus algorithm, in 2015. His research interests include blockchain, big data, and FinTech.

Dr. Liao was a recipient of the Google's Highest Award, Founders' Award for his contributions to Android.