Journal of Physics: Conference Series

PAPER • **OPEN ACCESS**

Deep Convolutional Neural Network for Air Quality Prediction

To cite this article: Yushun Mao and Shiejue Lee 2019 *J. Phys.: Conf. Ser.* **1302** 032046

IOP Conf. Series: Journal of Physics: Conf. Series **1302** (2019) 032046 doi:10.1088/1742-6596/1302/3/032046

Deep Convolutional Neural Network for Air Quality Prediction

Yushun Mao and Shiejue Lee

Department of Electrical Engineering, College of Engineering, National Sun Yat-Sen University, 70, Lien-hai Rd., Kaohsiung 80424, Taiwan ROC.
Email: m073010024 @student.nsysu.edu.tw, leesj@mail.ee.nsysu.edu.tw

Abstract. In this paper, we tackle air quality forecasting by using deep learning approaches to predict the hourly concentration of air pollutants (e.g., ozone, particle matter PM2.5 and sulfur dioxide). Deep learning (DL), as one of the most popular techniques, is able to efficiently train a scalable model on big data by optimization algorithms. The model is trained for air quality prediction with time series data. Our method takes the deep convolutional neural network (CNN) as the sequence module and inputs the time series data into the CNN model in turn for training. CNN is composed of many functional layers, such as convolution, pooling and ReLU. Convolution layer can effectively extract the sequential features of time series data. Sequential features work better than general features of time series data. Down-sampling in CNN is performed by the Pooling layer. Experimental results show that CNN performs well for air quality prediction.

1. Introduction

Air quality prediction is an important issue, as economy is improved; air pollution is growing worse and can produce health risk to human beings. For example, diesel oil and industrial emissions are the main factors that cause cancer risk (70%), in central Tehran [1]. The air we breathe is full of exhaust gases and pollutants from motor vehicles and industrial emissions and coal-fired power plants, including respirable suspended particulates (PM10, PM2.5), nitrogen dioxide (NO2), ozone and sulfur dioxide. With particle diameter of less than 2.5 micrograms, PM2.5 has got special attention by the medical profession because of the fine particles into the nasal cavity. In addition to cause rhinitis, fine particles also pass along the respiratory tract into the lungs, affecting cardiovascular health, increasing the chance of chronic lung disease, cardiovascular disease and the risk of lung cancer, and even the risk of early death. Medical science has long confirmed that PM2.5 has a significant relationship with lung cancer risk. Particulate matter usually comes from diesel vehicles and industrial exhausts, which contain Polycyclic Aromatic Hydrocarbons. In addition, PM2.5 can stimulate the inner wall of blood vessels and clot coagulation after it invades the body, which may lead to cardiovascular embolism and heart disease. Therefore, living in an environment with high PM2.5 concentration for a long time will increase the risk of early death due to heart disease and chronic lung disease [2].

Traditionally, the air pollution prediction (AQP) problem is generally regarded as one of the concrete applications of time series prediction. Conventional methods, such as nonlinear regression, and linear regression have been widely used in AOP. Regression analysis is rapid in modeling, and simple causality analysis is quite effective for small data sets.

However, it is difficult to model nonlinear data or to analyze data sets with high complexity. Neural networks can effectively solve the above shortcomings and can more fit the problem of air prediction for large high-complexity data sets. SVR [3] is extended from support vector classification (SVC) [4], using kernel functions to map data to a high-dimensional space. Recently, convolutional neural network (CNN) and recursive neural network (RNN) are playing a more and more important role in time series prediction, e.g., machine translation, speech recognition, text detection, financial prediction and other fields related to time series. In particular, CNN is widely used in face recognition, automatic driving, object detection and image processing. The biggest advantage of CNN lies in that it has various types of units, such as convolution, pooling, linear rectification function, hyperbolic function, normalization and regularization. Different processing units can generate various local characteristics and bring different processing effects for time series signals. The deep structure allows a variety of different processing units to be stacked on top of each other, so that the deep learning model can represent the sequential characteristics of time series signals with different dimensional vectors.

The purpose of this paper is to predict air quality for the future 48 hours using a convolutional neural network. In Section 2, our proposed air quality prediction model is described. Experimental results are presented in Section 4, and Section 5 gives a conclusion.

2. Air Quality Prediction Model

Our proposed AQP model is mainly based on CNN network which consists of layers of convolution and pooling, plus a fully connected layer, as shown in Figure 1.

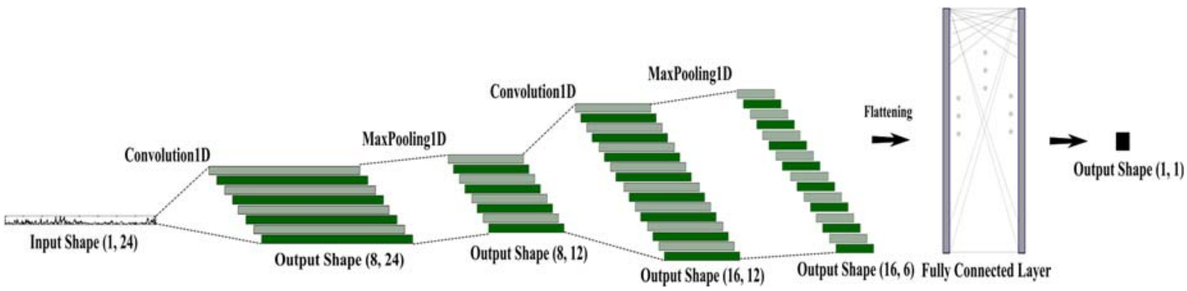

Figure 1. Illustration of the CNN architecture.

2.1. Convolution Layer

There are some important concepts to be mentioned. Kernel vector is the most important in a convolution layer, and it is also called the sliding window. Convolution operates through the kernel vector which slides sequentially on the input vector. The kernel vector is multiplied element-wise with the input vector, and the sum added by a bias is put through the activation function to form the feature map for the next layer [5]. The one-dimensional convolution of the data I of the previous layer with filter k is given by

$$\mathrm{C}_{\mathrm{p}}^{\mathrm{l}} = \sigma\left(\mathrm{I} * \mathrm{k}_{\mathrm{l,p}}^{\mathrm{l}} + \mathrm{b}_{\mathrm{p}}^{\mathrm{l}}\right) \tag{1}$$

$$\mathrm{C}_{\mathrm{p}}^{\mathrm{l}}(\mathrm{i}) = \sigma\left(\sum_{\mathrm{u}=-\mathrm{U}}^{\mathrm{U}} \mathrm{I}(\mathrm{i}-\mathrm{u}) \cdot \mathrm{k}_{\mathrm{l,p}}^{\mathrm{l}}(\mathrm{u}) + \mathrm{b}_{\mathrm{p}}^{\mathrm{l}}\right) \tag{2}$$

With filter k as

$$k = [k_{-U} \quad \cdots \quad k_0 \quad \cdots \quad k_U]$$

where $\sigma(\ldots)$ is the activation function, b_p^l is the bias for the output of convolution, p is number of feature maps (the number of filters) on the lth layer, U denotes the range of indexes of filter k, $*$ denotes the convolution, and i is the index of the feature map. The activation function can be defined as

$$\textit{Sigmoid: } \sigma(\mathrm{x}) = \frac{1}{1+\exp(-\mathrm{x})} \tag{3}$$

$$\textit{Tanh: } \sigma(\mathrm{x}) = 2 * \mathrm{sigmoid}(2\mathrm{x}) - 1 \tag{4}$$

$$\textit{ReLU: } \sigma(\mathrm{x}) = \max(0, \mathrm{x}) \tag{5}$$

2.2. Pooling Layer

The key concept of the pooling layer is to provide translational invariance, aiming to preserve the detected features in a smaller representation by discarding less significant data at the cost of resolution [6]. The pooling of previous convolution layer's feature maps with window size of s can be defined as

$$S_p^l(i) = \max\left(\sum_{u=0}^{s-1} C_p^l(2i-u)\right) \tag{6}$$

$$S_p^l(i) = \mathrm{avg}\left(\sum_{u=0}^{s-1} C_p^l(2i-u)\right) \tag{7}$$

Max pooling operates by sliding the window sequentially on the input vector, and then takes the maximum value of the window region and gets rid of the other values. Average pooling is done, instead, by taking the mean of the values in the window region.

2.3. Fully Connected Layer

Fully Connected layer contains one to three layers, and it is usually a Multilayer perceptron (MLP) which is a kind of feedforward neural networks. A MLP basically consists of three layers of nodes, input layer, output layer and hidden layer. Except for the input nodes, the output of each node is connected to the activation function. The activation function is Linear or nonlinear, as those defined in Eqs. (3), (4) and (5). The weights and biases associated with a MLP are usually optimized by applying the backpropagation algorithm for training. The output of the MLP is given by:

$$Y_p^l = \sigma(W_p^l \cdot F_p^l + b_p^l) \tag{8}$$

Where W_p^l is the weight matrix between the hidden layer and the output layer, F_p^l is the output of the hidden layer, b_p^l is the bias, and σ is the activation function associated with the output layer.

2.4. Proposed CNN Architecture

Our proposed CNN architecture for AQP is shown in Figure 2. Since the data for AQP are time-dependent, they are time-series data. Therefore, all the components in Figure 2 are mainly one-dimensional vectors. For example, all the kernels are vectors and all the feature maps are vectors. The input to CNN is a 24-vector, comprising 24 concentration values of the air pollutant being considered.

Input vectors to CNN are taken as follows. The 24 values taken at 24 consecutive time steps form the input vector. For training data, the value at the next step is taken as the desired output. For example, suppose we have 100 values taken sequentially in time. The 1st through 24th values form the input vector and the 25th value is taken as the desired output; the 2nd through 25th values form the input vector and the 26th value is taken as the desired output; etc. To predict the value at the kth time step, the 24 values at the previous time steps are taken as the input vector.

All layers of the CNN are grouped into four parts. Part one mainly deals with convolution. The input or the output of the previous layer is convolved with the kernel by Eq. (1) to extract the sequential features of time series data. The results are then transformed by the ReLU activation function into a positive linear form, as shown in Eq. (5). Part two is concerned with max pooling to get the maximums of feature maps over a 1×2 sliding window. Part three concatenates 16 vectors together to form a long vector with the length of 1×2×16=32. Part four is a two-layer MLP, defined as the output layer in the CNN architecture, as shown in Eq. (8).

2.5. Deep Learning Frameworks

Machine learning is developing at an incredible speed. The key is to deploy the machine learning model to work properly in embedded devices, thus making applications smarter. As of today, deep learning has been widely used in business operations, and has been proven to have excellent results. Research on artificial intelligence makes it possible to solve complex problems.

Deep learning is a complex task. For scientists and engineers, it is a difficult challenge to successfully build feasible mathematical models and deploy them in embedded devices. Today, some well-known e-commerce companies have released Deep Learning frameworks, such as Google/Tensor Flow, Amazon/MXNet, Facebook/PyTorch, Baidu/PaddlePaddle, Alibaba/ x-deep learning and so on, for researchers to use. These frameworks allow us to easily build complex mathematical models while simplifying difficult programming challenges. In this paper, we use PyTorch to build our CNN Model.

Torch is a scientific computing framework that offers wide support for machine learning algorithms. It is used widely amongst industry giants such as Facebook, Twitter and Google. It employs Compute Unified Device Architecture (CUDA) along with C/C++ libraries for the processing and was basically made to scale production of building models and overall flexibility. PyTorch is basically a port to Torch, and it runs on Python, which means that anyone with basic understanding of Python can get started on building their own deep learning models [7].

3. Experiments

In this section, we present some experimental results to show the effectiveness of our proposed prediction model. PM2.5 data are used in our experiments. Similar models can be used to predict the concentration of other pollutants.

3.1. Datasets

We collected air quality and meteorological data every hour from 77 locations from the database of Environmental Protection Administration, Executive Yuan [8].

There are 15 attributes associated with the dataset. The values of attributes were observed and recorded. The 15 attributes are SO_2, CO, O_3, PM_{10}, $PM_{2.5}$, NOx, NO, NO_2, THC, NMHC, CH_4, WIND_SPEED, WS_HR, AMB_TEMP, RH.

3.2. Loss Function and Ground Truth

The predicted PM2.5 values are compared with ground truth results obtained at each location and the root mean square error (RMSE) is adopted to evaluate prediction performance:

$$RMSE = \sqrt{\frac{\sum_1^n (\widehat{y_i} - y_i)^2}{n}} \tag{9}$$

Where $\widehat{y_i}$ and y_i are the predicted and ground truth-values, respectively, for the *i*th hour, and *n* is the number measurements within a batch size. We calculate the RMSE error for each batch (batch size = 32), and then compute an average of RMSE for each epoch. Lower error indicates higher prediction accuracy.

3.3. Experimental Settings

In this paper, we use RMSprop algorithm [9] to update the weights of filters and of the fully connected layer in the CNN. The parameters are chosen as p = 8 (for Conv1) and 16 (for Conv2), β = 0.99, ε = 1×10^{-8} (term added to the denominator to improve numerical stability). We will discuss the effect of max pooling and average pooling, and of different kernel sizes, on performance. RMSprop equations are given as

$$s_{dw} = \beta s_{dw} + (1 - \beta) dW^2 \tag{10}$$

$$s_{db} = \beta s_{db} + (1 - \beta) db^2 \tag{11}$$

$$W = W - \alpha \frac{dW}{\sqrt{s_{dw}} + \varepsilon} \tag{12}$$

$$b = b - \alpha \frac{db}{\sqrt{s_{db}} + \varepsilon} \tag{13}$$

3.4. Performance and Discussion

The results of the proposed different CNN models on Air Quality Monitoring dataset is shown in Table 1. RMSE is used to evaluate the performance of the models. In this table, the results of the two proposed CNN models, with max pooling and average pooling, respectively, are listed. The numbers 1, 3, and 5 denote the kernel size used. The best performance is highlighted in bold in the table.

Table 1. RMSE RESULTS

	CNN-Max-1	CNN-Max-3	CNN-Max-5	CNN-Avg-1	CNN-Avg-3	CNN-Avg-5
Train	9.17	**6.07**	6.22	8.85	6.43	**6.11**
Test	9.72	8.20	**7.25**	9.49	8.31	**7.37**

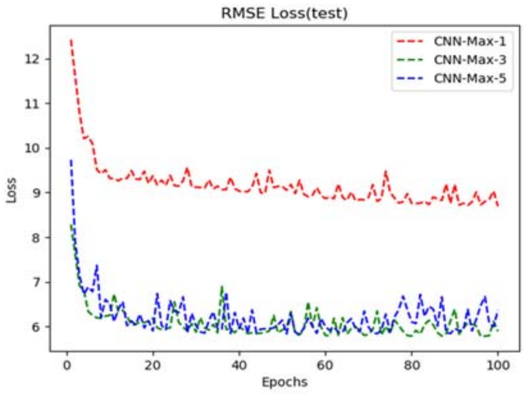

Figure 2. Performance of CNN with max pooling

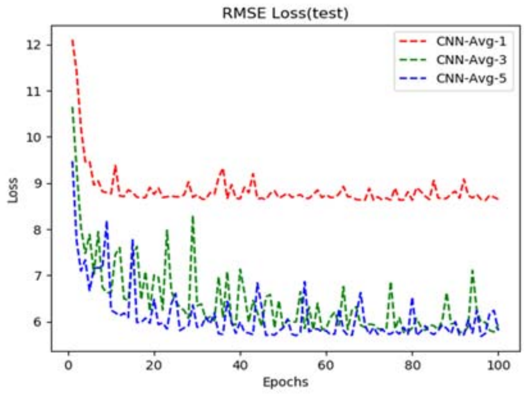

Figure 3. Performance of CNN with average pooling

Performance of CNN with average pooling is shown in Figure 4. CNN with kernel size = 5 has the best performance in Table 1. It can be seen from Table 1 that the CNN with average pooling is better than the CNN with Max pooling, although the differences are not too much.

4. Conclusions

We have described how CNN, one of deep learning approaches, is used to predict the hourly concentration of air pollutants. The model is trained for air quality prediction with time series data. CNN is composed of many functional layers, such as convolution, pooling and ReLU. Convolution layer can effectively extract the sequential features of time series data. Experimental results have shown that CNN performs well for air quality prediction.

5. Acknowledgments

The authors are grateful to the Taiwan Environmental Protection Administration for providing the data used in this study.

6. References

[1] Sina Taghvaee, Mohammad H.Sowlat, Mohammad Sadegh Hassanvand, Masud Yunesian, Kazem Naddafi, Constantinos Sioutas, Source-specific lung cancer risk assessment of ambient PM2.5-bound polycyclic aromatic hydrocarbons (PAHs) in central Tehran, 2018, abstract.
[2] Wen-Chi Pan, Chih-Da Wu, Mu-Jean Chen, Yen-Tsung Huang, Chien-Jen Chen, Huey-Jen Su, Hwai-I Yang, "Fine Particle Pollution, Alanine Transaminase, and Liver Cancer: A Taiwanese Prospective Cohort Study (REVEAL-HBV)," JNCI: Journal of the National Cancer Institute, Volume 108, Issue 3, 1 March 2016, djv341.
[3] Vladimir Vapnik, The Nature of Statistical Learning Theory, Springer New York, NY, 1995.
[4] Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik, "A training algorithm for optimal margin classifiers," In Proceedings of the Fifth Annual Workshop on Computational. Learning Theory, pp. 144-152, ACM Press, 1992.
[5] Zhifei Zhang, University of Tennessee, Knoxville, TN, America, "Derivation of Backpropagation in Convolutional Neural Network," 2016, unpublished.

[6] Environmental Protection Administration Executive Yuan R.O.C.(Taiwan) environmental resources database.
[7] Jeff Hale, Deep Learning Framework Power Scores 2018, unpublished.
[8] Environmental Protection Administration Executive Yuan R.O.C. (Taiwan) air quality Monitoring website.
[9] Geoffrey Hinton, Neural Networks for Machine Learning, slides of Neural Networks for Machine Learning:Lecture 6e rmsprop:Divide the gradient by a running average of its recent magnitude, Coursera, 2016.