Predictive model for Air Pollution using Neural Networks

Samuel Cheung
Big Data Technology
Hong Kong University of Science and Technology
Clearwater Bay, N.T., Hong Kong
ckcheungao@ust.hk
20476499

ABSTRACT

Air pollution problem in urban cities are becoming more serious in recent years. Sometimes the air pollutant may reach a level that cause health concern to citizen to do outdoor activities. Take Hong Kong as example, the Environmental Protection Department launched the Air Quality Health Index to indicate the current and forecast the health risk level.

In this project, we built two predictive models for air pollution using neural networks. One model is LSTM while another one is the custom designed recurrent neural network (RNN) model. We trained the two models with time series data of pollution and meteorological data collected from 35 air quality stations in Beijing. Our result shows the models produce a high accuracy prediction for the next time period. Our model also extends this prediction from single time period to 2 days (48 hours) in the future.

Categories and Subject Descriptors

Theory of computation: Machine learning theory
Computing methodologies: Machine learning algorithms, Neural networks

General Terms

Algorithms, Documentation, Design, Theory

Keywords

Deep Learning, Neural networks, Convolution Neural Network (CNN), Recurrent Neural Network (RNN), Long Short Term Memory Network (LSTM), Predictive Model

1. Theoretical Background

There are thousands of artificial neural networks proposed by researchers. Some are whole new approaches while some are modifications to existing approaches. In general, there are three classes of artificial neural networks:

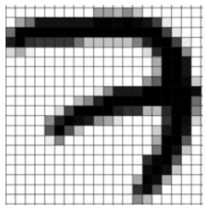
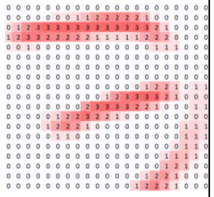
- Multilayer Perceptrons (MLP)
- Convolutional Neural Networks (CNN)
- Recurrent Neural Networks (RNN)

We will only focus on CNN and RNN.

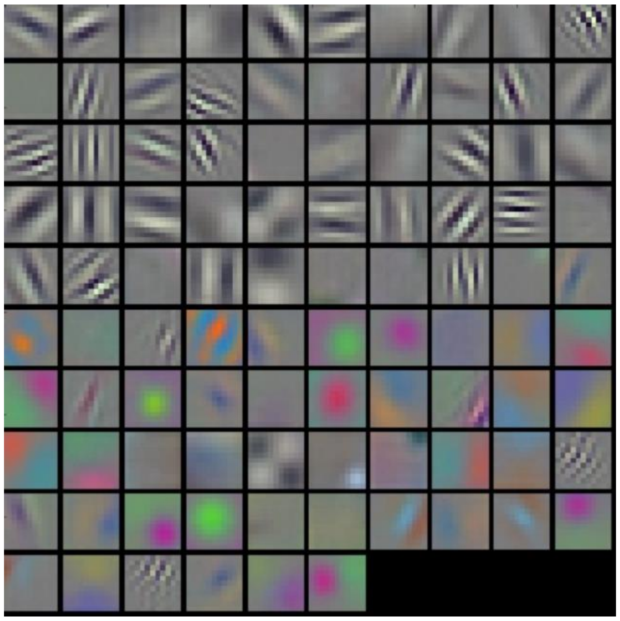
1.1 Convolutional Neural Networks (CNN)

Traditional feedforward neural network requires a 1d input weights. It has difficulties to deal with the problem that the input has spatial relationship. Flattering the image from pixel matrix to long vector of pixel values will lose the spatial structure in the image [1].

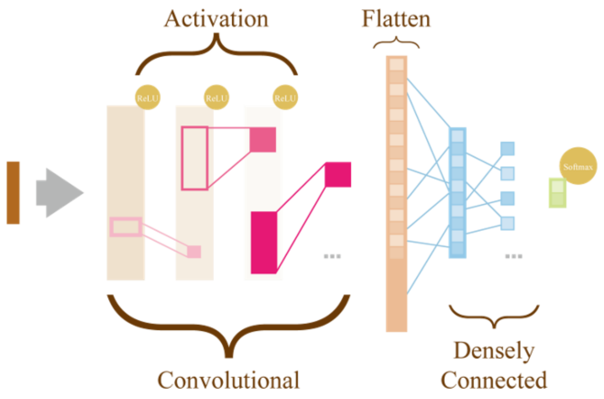
CNN overcomes this issue by learning internal features using small squares of input data. For example, a filter for detecting horizontal lines is applied to the image. The area with horizontal line will have large activation value while other activation values are small.

Input	**Filter**	**Output**
	$\begin{array}{\|c\|c\|c\|}\hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$	

In the convolution layer (conv2D), a bunch of filters were applied. Each feature is learnt from one filter.

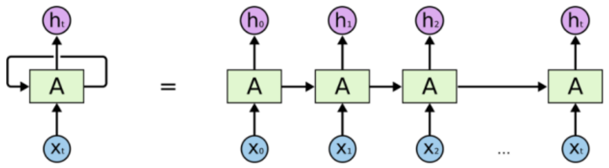


After some pooling layers and fully-connected layers, the image is mapped to output variable.



1.2 Recurrent Neural Networks (RNN)

Unlike the standard feed-forward networks, RNN has an additional loop in the architecture [2]. So recurrent networks have two sources of input, the present and the recent past. This connection add memory to the network and allow RNN to learn broader abstractions from the input sequence.



RNN works well in the situation where the gap between the relevant information and the prediction is small. When this gap grows, RNN is no longer able to connection the information from the very past to the current prediction.

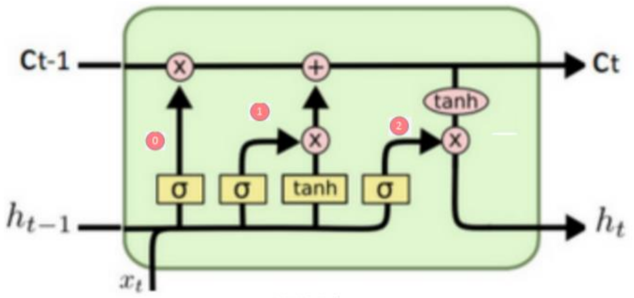
1.3 Long Short Term Memory (LSTM)

The short-coming of RNN cannot handle long-term dependencies is addressed by LSTM. The key idea is to use memory cells and gates:

$$\mathbf{c}_{(t)} = \mathbf{f}_t \otimes \mathbf{c}_{(t-1)} + \mathbf{i}_t \otimes \mathbf{a}^{(t)}$$

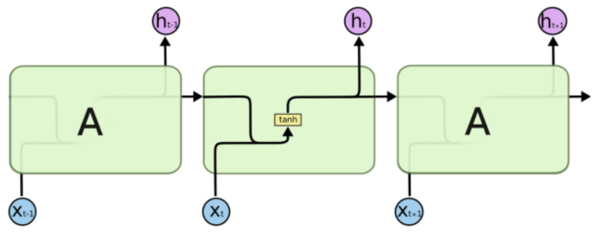
$\mathbf{c}_{(t)}$ is the memory state at t; $\mathbf{a}^{(t)}$ is the new input at t.

The forget gate $\mathbf{f}_t$ (range from 0 to 1) is controlling whether the current memory is kept while the input gate $\mathbf{i}_t$ with the same range is controlling if the current value in the memory cell is replaced by the new input.

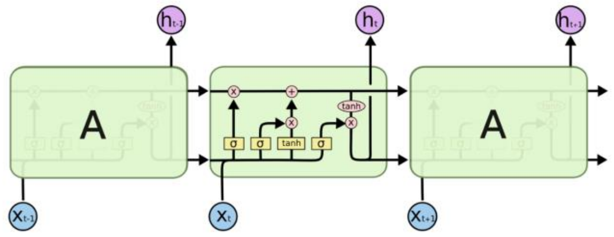


LSTM Cell

Below diagram shows a standard RNN contains a single neural network layer.



While comparing to LSTM, LSTM contains four interacting layers.



The main reason that LSTM can learn longer term dependencies is due to the horizontal line which pass through the top of the diagram. This line runs through the entire chain where useful information can flow through without losing details.

1.4 Our Approach

Spatial relationship

In order to address the spatial relationship between air stations and the grid weather. We translate the spatial distribution of air pollution and weather data into image-like maps. We use CNN to extract features from these maps.

Temporal relationship

On the other hand air pollution and weather data is the time series data where no doubts RNN or LSTM is a go-to neural network architecture to use.

Some previous studies [3], [4] supports our ideas as well. We experimented models using combinations of these algorithms.

2. Data

2.1 Data Description

The data were collected from three sources.

The air quality data which contains the concentration of PM2.5 (ug/m3), PM10 (ug/m3), NO2 (ug/m3), CO (mg/m3), O3 (ug/m3) and SO2 (ug/m3) collected from 35 air quality stations across Beijing.

The observed weather which contains weather, temperature, pressure, humidity, wind speed, wind direction collected from 19 weather stations.

The grid weather data is a continuous data distribution over earth surface derived from observed weather data through a complicated calculation algorithm.

We use the hourly air quality and weather data in the period between Jan 2017 and Apr 2018 to train the model. We then make use the trained model plus the hourly weather data on 1st May and 2nd May to predict the 48 hours air quality on these 2 days.

2.2 Data Pre-processing and Feature

As the air quality data updates frequently that some of the observations in the middle are missing. Dealing with the missing values is a crucial step before the further of feature engineering. On treating those missing values, we divide the process into two parts: Cluster-based Filling and Forward-Backward Filling.

2.2.1 Data Preprocessing

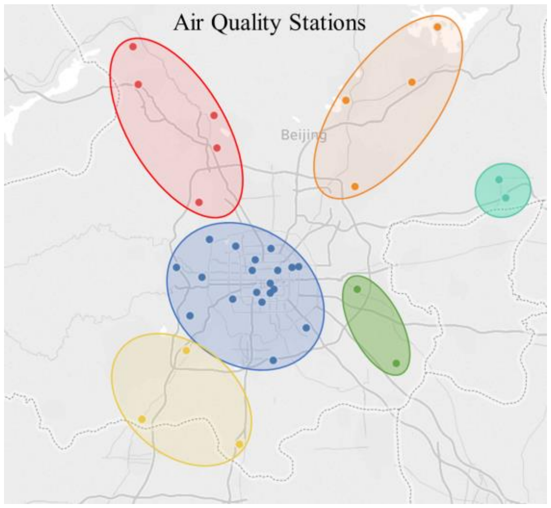
I. Cluster-based Filling

For the large air quality datasets with missing values, complicated methods are not practical because of the high computation cost. We tend to find a simple solution to our problem. Instead of using mean or forward-backward filling on the whole dataset of air quality data, we use mean in its subset obtain by clustering.

With the existing longitude and latitude of weather stations, we are here to utilize their geolocation for clustering.

K-mean Clustering is our choice applying on this problem aiming to partition the air quality stations into k clusters. To ensure that we can fill the missing elements based on the clustering result, it is necessary to ensure that each cluster contains at least 2 air stations.

Together with the evaluation metric of silhouette score and the mentioned restriction on each cluster, we choose k = 6 for the optimal clustering number. The figure below then shows the clustering result:

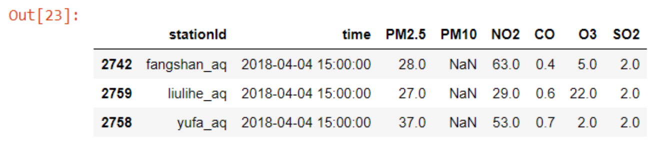


Based on the clustering result, the missing values of the stations then would be fill at each time step per each cluster.

II. Forward-Backward Filling

As shown in the previous figure concerning about the clustering, one can easily observe that some of the cluster only contains 2-3 air stations. Therefore, there are certain time steps that all the stations with in the cluster has missing values in specific attribute.

Here is an example that PM10 value is missing within the cluster of only three stations.



```
Out[23]:
```

	stationId	time	PM2.5	PM10	NO2	CO	O3	SO2
2742	fangshan_aq	2018-04-04 15:00:00	28.0	NaN	63.0	0.4	5.0	2.0
2759	liulihe_aq	2018-04-04 15:00:00	27.0	NaN	29.0	0.6	22.0	2.0
2758	yufa_aq	2018-04-04 15:00:00	37.0	NaN	53.0	0.7	2.0	2.0

To deal with those value, we first filter the whole processed dataset station by station and sort the timestamp in ascending order. Then, we fill forward or backward based on the past or future feature values of each station.

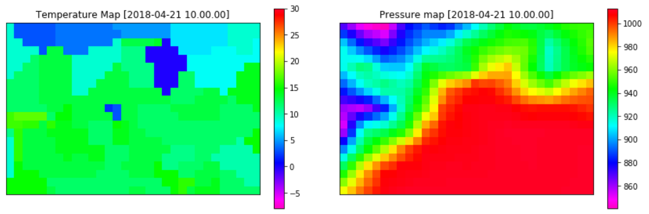
2.2.2 Feature Engineering

Feature Engineering is the process of creating features from the given data that cope with the machine learning algorithm. For simplicity, although the dataset has provided the observed weather data, majority of the feature engineering process is conducted by the grid weather data instead.

For the attempt of different machine learning model, here we manipulate the grid weather data in two different ways:

<u>Spatial relationship construction</u>

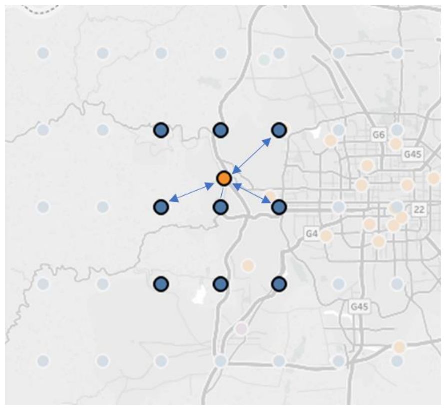
The grid weather data and air quality data is mapped to the coordinate of each grid and air station. For example, a linear data sequence (651 grids) is then transformed into 2-dimensional grid map (21x31). One map is produced for each feature. At the end, the dimension of the weather data is 21x31x5 for every hour. The weather map can be easily visualized by plotting the map.



<u>Temporal relationship construction</u>

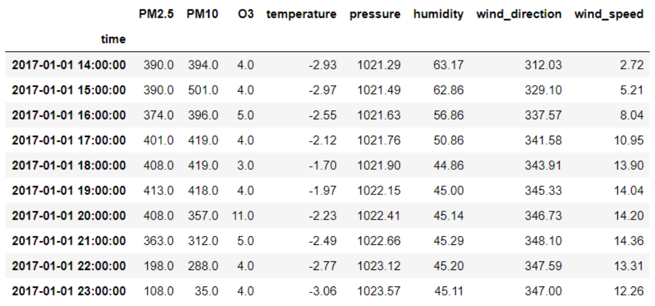
Due to the absence of weather information (e.g temperature, pressure) within the air quality data, we here make use of the geolocation of grid and air quality stations to combine two sets of data together.

As shown in the below figure, the air pollutant data within the air station (represented by orange dot) would be combined with the weather data of the nearest grid point.



As a result, we store the joined table for each station in a csv file. The table below shows the content of the csv file after merging.

Table : Merged Data at dongsihuan_aq

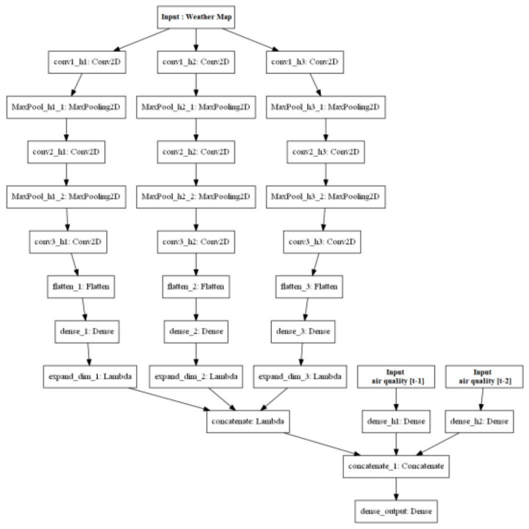


time	PM2.5	PM10	O3	temperature	pressure	humidity	wind_direction	wind_speed
2017-01-01 14:00:00	390.0	394.0	4.0	-2.93	1021.29	63.17	312.03	2.72
2017-01-01 15:00:00	390.0	501.0	4.0	-2.97	1021.49	62.86	329.10	5.21
2017-01-01 16:00:00	374.0	396.0	5.0	-2.55	1021.63	56.86	337.57	8.04
2017-01-01 17:00:00	401.0	419.0	4.0	-2.12	1021.76	50.86	341.58	10.95
2017-01-01 18:00:00	408.0	419.0	3.0	-1.70	1021.90	44.86	343.91	13.90
2017-01-01 19:00:00	413.0	418.0	4.0	-1.97	1022.15	45.00	345.33	14.04
2017-01-01 20:00:00	408.0	357.0	11.0	-2.23	1022.41	45.14	346.73	14.20
2017-01-01 21:00:00	363.0	312.0	5.0	-2.49	1022.66	45.29	348.10	14.36
2017-01-01 22:00:00	198.0	288.0	4.0	-2.77	1023.12	45.20	347.59	13.31
2017-01-01 23:00:00	108.0	35.0	4.0	-3.06	1023.57	45.11	347.00	12.26

3. Models

3.1 CNN + DNN Model

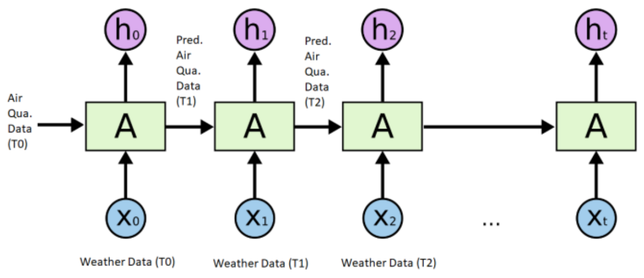
Convolutional Neural Networks have become the de facto algorithm in the field of computer vision. Here we first apply the CNN model to the image-like input of the grid data with size of 21 x 31 and 6 channels at the previous hour. The other inputs are air pollution data in the past two hours which would be passed through a simple DNN model. Those input has the shape of (3,35) corresponding to the three target features and total 35 air stations. As the output layer, we stack both of the model to output the air pollutant value in the current predict hour. The model can be found in the Spatial_Model.py with the spatial_weight.h5 being trained. Below is a graph of its architecture :



3.2 LSTM Model

In our LSTM model construction, an assumption that the air pollution data is persistent over a short period of time is made. i.e. the predicted air pollution level in the next hour will not be very different to the previous observed air pollution level.

The predicted air quality data together with the observed weather data in the next time frame will be fed into the LSTM model for the next prediction. The process continues until we complete the whole prediction period.



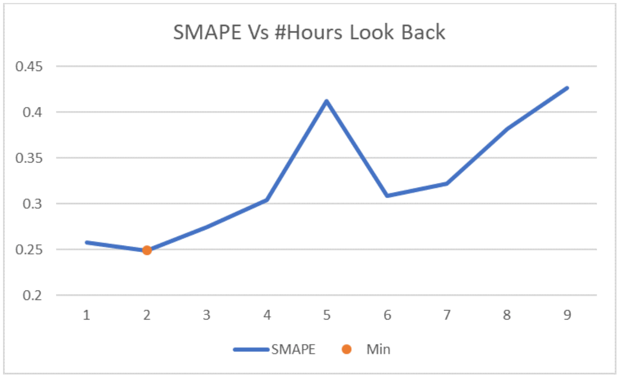
Our LSTM model has 1 LSTM layer and 1 dense layer.

```
________________________________________________________________
Layer  (type)                      Output  Shape         Param  #
================================================================
lstm_1  (LSTM)                      (None,  100)           46800
________________________________________________________________
dense_1  (Dense)                     (None,  3)              303
================================================================
Total                           params:                   47,103
Trainable                         params:                 47,103
Non-trainable                          params:                 0
________________________________________________________________
```

4. Experiments and Results

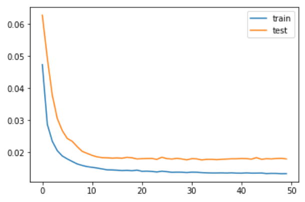
4.1 Experiment on fine tuning the period of historical time series data fed into the models

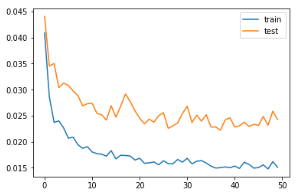
From our experiment, the best number of hours for the model look back previous observed data is 2 hours. If we pass 9 hours air pollution data into training model, the SMAPE has large volatility. It makes sense that air pollution data and weather 9 hours before may have little correlation to the air pollution level in the next hour. Instead, using 2 hours history data can obtain a smooth diminishing SMAPE curve.



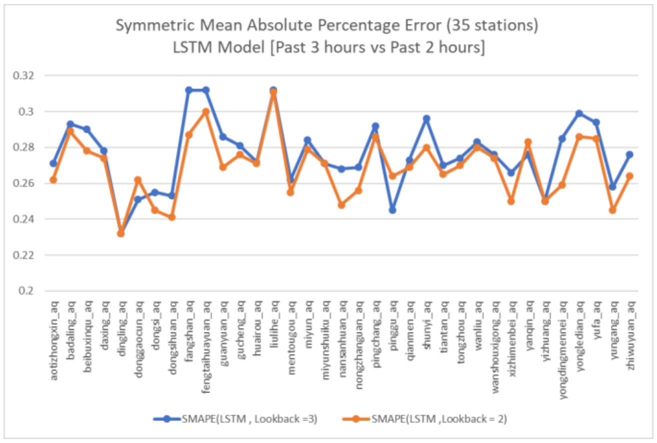
MAE Loss - 2 Hours

MAE Loss - 9 hours





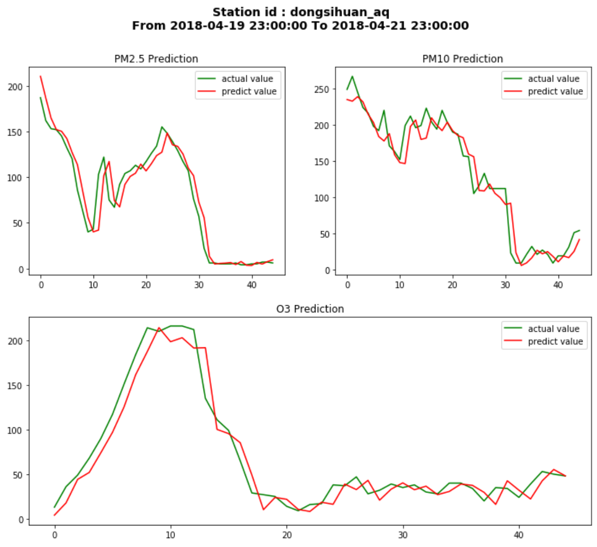
We validate our above observation across all the 35 air stations. It is proven that using 2 hours historical data consistently perform better than 3 hours (and also other number of hours) historical data.



We applied this 2 hours historical data setting into both of our two models training.

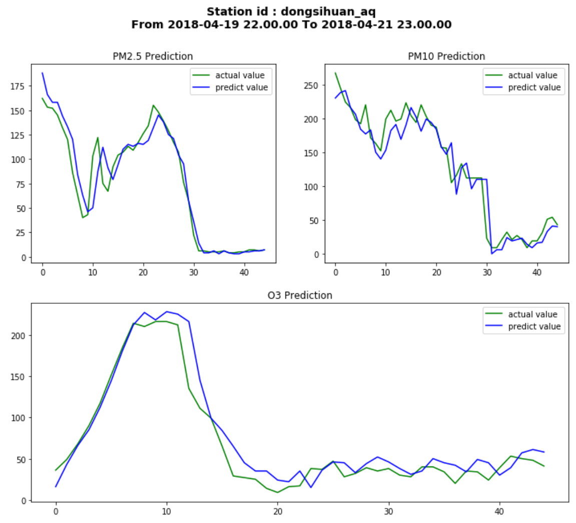
4.2 LSTM Result

We picked one station to illustrate the LSTM result comparison of PM2.5, PM10 and O3 between actual value and predicted value in the testing period.



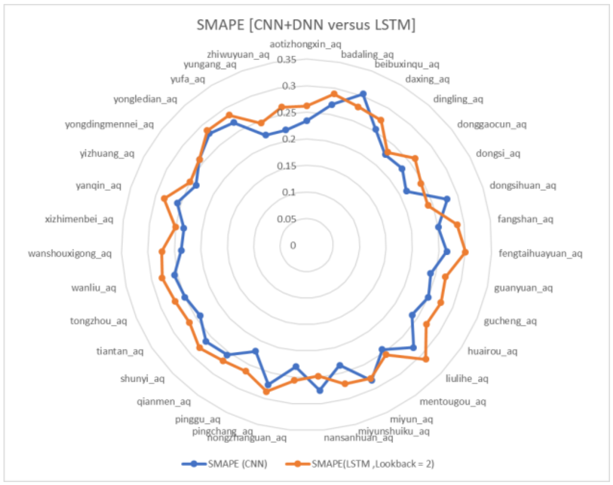
4.3 CNN + DNN Result

Similar to LSTM, by choosing the same air quality station. the following are the actual and predicted value illustration using CNN+RNN model.



4.4 Result comparison between two models

Only with few exceptions, our customized CNN+DNN model constantly achieve smaller SMAPE than LSTM model.



The average error of both models across all 35 air quality stations.

	CNN + DNN	LSTM
SMAPE	0.251	0.269
RMSE	25.86	28.33

5. Conclusions and Future Work

From the test result, both of the models are doing quite well in predicting air pollution in the near future hours. The graph shown in the LSTM result in the previous section demonstrated the predicted result curve is just like the actual result curve with a time delay. The key of better fine tuning LSTM model is to reduce this time delay as much as possible. CNN+RNN model result graph shows a more complex behaviour. The prediction result of some period is as accurate as the actual without any time delay, while some is more volatile. A deeper analysis is required to have a better understanding and explanation of the CNN+RNN result.

As of the future work, below are some highlights of the focus areas. On top of the convolutional neural network, the exploration of convolutional structure combining with the time series model such as convolutional recurrent neural network is encouraged, as we can best fit our image-like weather mapping into the model. For the prediction output, as our proposed approaches are based on the past hours input to predict the air pollutant values in the next hour. For 48-hours prediction output, we have to iterate the next prediction based on last prediction values. As a result, the error of the long-term prediction output may grow exponentially due to the combination of errors in the past predictive values. To address the problem of growing error, the exploration of other model with output in a customized time range (i.e. 24 to 48 hours Output) is a must. One proposed approach is that the first step is to use the LSTM or RNN model on predicting the average pollutant value daily and the second step is to use the CNN or CRNN to adjust the precise value in each hour on that day based on the given prediction from the first part.

6. Reference

[1] https://machinelearningmastery.com/when-to-use-mlp-cnn-and-rnn-neural-networks/

[2] http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[3] C. J. Huang, P. H. Kuo, A Deep CNN-LSTM Model for Particulate Matter (PM2.5) Forecasting in Smart Cities, 2017

[4] V. Duc Le, S. K. Cha, Real-time Air Pollution prediction model based on Spatiotemporal Big data, 2018.

[5] V. Reddy, P. Yedavalli, S. Mohanty, Deep Air: Forecasting Air Pollution in Beijing, China, 2017

[6] Xiao Feng, Qi Li, J. Hou, L. Jin, J. Wang, Artificial neural networks forecasting of PM2.5 pollution using air mass trajectory based geographic model and wavelet transformation, 2015

[7] S. Mourtzinis, Juan I. Rattalino Edreira, S P Conley, P. Grassini, From grid to field: Assessing quality of gridded weather data for agricultural applications, 2016