Air concentration prediction based on LSTM

yuntao, Zhang
yzhangii@connect.ust.hk
20551081

Abstract
In this report, I use LSTM neutral networks combined with k-nearest neighbor algorithm to predict the hourly concentration levels of three pollutants, PM2.5, PM10 and O3 on May 1 and May 2 for 35 stations in Beijing, China. The prediction is based on the hourly air quality and weather information at the 35 stations from January 1, 2017 to April 30, 2018.

Introduction

Air concentration prediction has been a popular topic in recent years since it plays a significant role in all aspects of air pollution control and planning. However, it is hard to predict the evolution of air pollutant due to its high variability caused by complex correlated factors. Neutral networks like Long Short-Term Memory (*LSTM*) recurrent neural networks are able to deal with multiple input variables almost seamlessly in time series problem, which is suitable for this project given numerous parameters of air quality and weather information. In this project, LSTM is used to model the problem. I also use meteorological parameters from variable length of time period before the date to predict the concentration. Then the prediction result is evaluated by symmetric mean absolute percentage error associated with mean squared error.

Data preparation

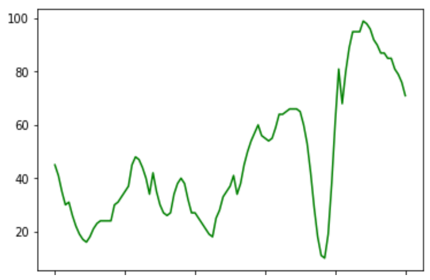
Original data files are composed of air quality or weather information from all the stations. In order to predict the air concentration from each distinct station, first I extracted the air quality of each air quality monitoring stations quality station from all the raw data files, then merge them into 35 files, each of which contains information of only one station in the total past time period, then check the air station location and get weather information at each air quality station by searching the corresponding longitude and latitude in the grid weather files, finally append weather information into the corresponding air station file generated before. The prediction of each station will be based on the corresponding generated file.

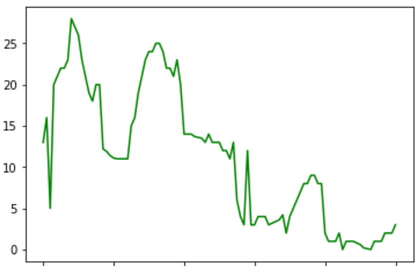
Since raw data contain large quantities of noises, missing values and even mistakes due to faults of measure instruments or human errors, data preprocessing is necessary. There are two steps for data preprocessing: processing null values and data noises reduction.

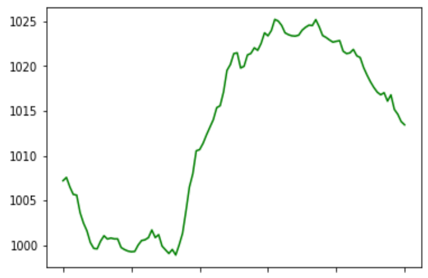
The most serious problem is null values contained in the raw data. The table below shows the rate of missing values in data of part stations. Some null values may last for a long period. For instance, air concentration values in the zhiwuyuan air quality station have been missing since November 24, 2017. To solve the problem, k-nearest neighbor algorithm is used to fill null values. In detail, euclidean distances between each pair of stations are calculated, then if any parameter is missing in one station, it will be filled by the mean of the parameter at the same time point from the k nearest stations. However, null values still exist because there are some parameters that are missed at all stations. Left null values may be dropped or filled by parameters at nearest time point.

station	yanqin	shunyi	zhiwuyuan	yufa	liulihe	qianmen	xizhimenbei
missing rate	0.120	0.124	0.255	0.139	0.194	0.130	0.132

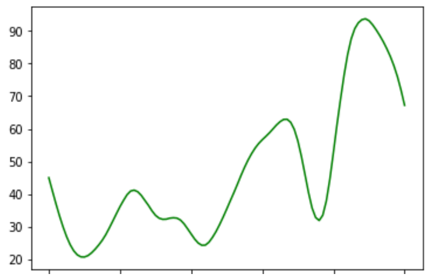
To estimate the degree of noises existing in the data, I extracted meteorological information from the same air quality station at different time and explore the relationship between the meteorological and time. Charts below show evolution of some of the meteorological parameters, the time period of which was selected in the condition of no missing values appearing so that the duration will be continuous and incomprehensible changes of air pollutants concentration can only be referred to data noises or mistakes.

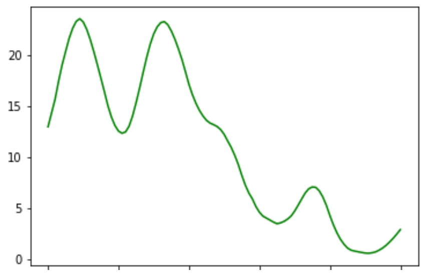

huimidity

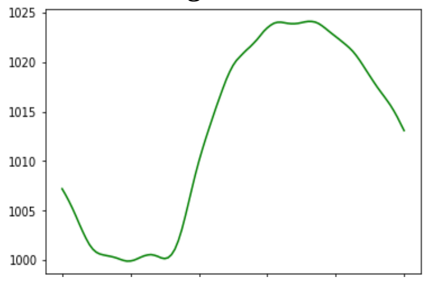

temperature


pressure

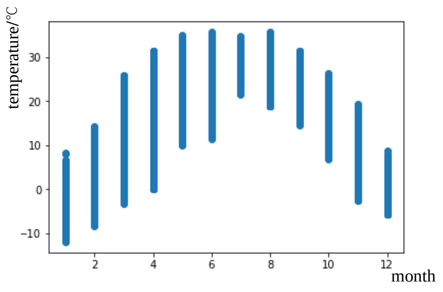
From figures above, it can be seen that since frequent fluctuation of parameters during the given continuous time period. I try to filter the time series by using filtfilt(), a filter function built in the scipy library to smooth the parameters during the time period during which no value is missing. The filter function can help remove noises without altering the phases of the time series. Figures below are the time series of above parameters after applying the filter function, which can be seen that the lines become smoother and basic value variations via time are preserved after filtering.


huimidity


temperature


pressure

Besides utilizing filter functions, I also checked whether there exists ridiculous variables such as negative values and abnormal temperature. The temperature changes via month at part of stations is illustrated by the following figure, it can be seen that temperatures are normally distributed and I didn't find any abnormal values.



The only improper values I found in raw data are the wind directions, which contain values of both 0 degree and 360 degrees at only two air quality stations. Then the value of 360 degrees is modified into 0 degree in the dataframes of the two stations.

After filling missing values and filtering data noises, I generated target labels by the shift function in pandas, that is, given the station and air type at which to be predicted, the column of the air type at the corresponding station is shifted up by k rows to generate a new target column, which means that the air concentration values are predicted by air quality and weather information k hours before.

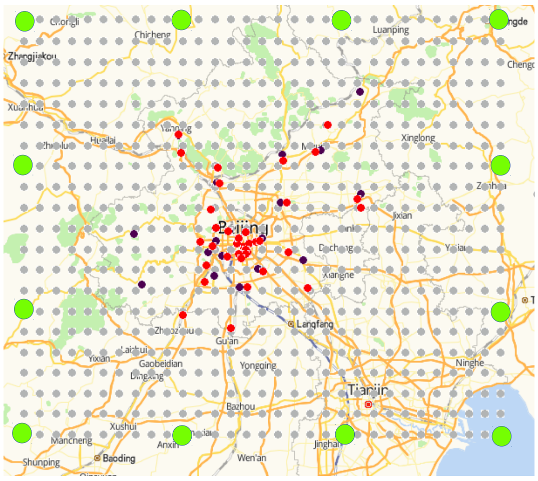
Feature engineering

Features play a significant role in model training. First I convert the time feature, which is string type, to three numeric type features: day, month and hour. Specifically the day denotes the week day rather than the day in the date whereas on weekends the traffic may be lighter, then the air concentration may be influenced by changes of air emission from vehicles or electronic devices.

For weather information, I extract humidity, wind speed, wind direction, pressure, temperature at the nearest grid from the station as weather information of each station. The features named weather was not adopted at first because there are only a month of data containing the weather feature and in other time the weather feature are just null values. However later it is found that the weather feature has strong relationship with the sudden mutation of the air concentration and brings a lower symmetric mean absolute percentage error on the test samples, which will be demonstrated in the evaluation section later.

I also add the time at which the air concentration is predicted like hour of the day, weekday and month. Other constructed features are rates of the target air concentration changes per hour, mean, maximum, minimum of the target air concentration at the same hour in the past 3, 7 days, latitude and longitude of the station and so on.

It can be found that all the above features are just combinations of the same location at different time. Since weather conditions at different locations in the same time, such as grids around the station also have a considerable impact on the air concentration, weather features at eight grids around the station are utilized in model training. Weather information in several outmost stations is also added into the feature space. The locations can be found in the following map, which are labeled with green dots.



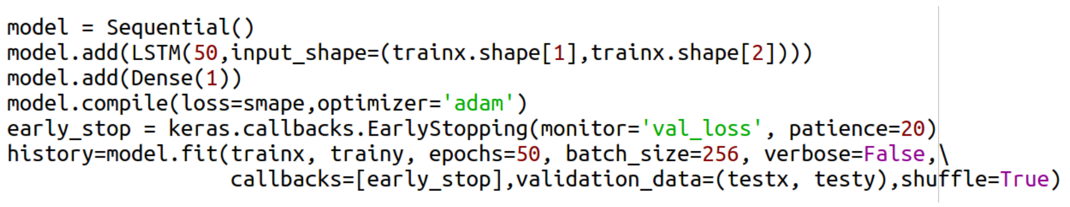
Afterwards correlations between different features and the target labels are calculated, the following table shows the correlations between part of features in "dongsi_aq" station and the concentration of PM2.5. No feature is abandoned because the absolute values of correlation of features are large enough and feature weight adjustment built in the neutral networks.

Feature	Correlation with PM2.5 concentration
humidity	-0.399506
temperature	0.104989
wind_direction	0.116214

CO	-0.294744
day	-0.132483
humidity at one outmost stations	-0.347036
wind_direction at one outmost stations	0.20588

Model Training and Evaluation

In this project LSTM are trained and evaluated on the symmetric mean absolute percentage error associated with mean squared error. LSTM (long short-term memory) neutral networks is the recurrent neutral networks which has a special structure and well-suited to make predictions based on time series. Moreover, since traditional linear methods such as linear regression can be difficult to adapt to multivariate problems, LSTM has great strength of handling such kind of problem. The structure and hyper parameters of neutral networks are shown below, which contains one LSTM layer and one dense layer. I also tried add one more dense layer between the two layers and compare the model performance.

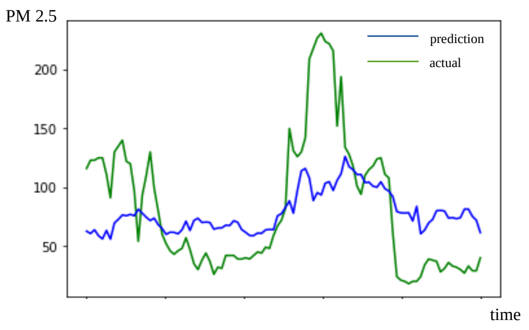


```
model = Sequential()
model.add(LSTM(50,input_shape=(trainx.shape[1],trainx.shape[2])))
model.add(Dense(1))
model.compile(loss=smape,optimizer='adam')
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=20)
history=model.fit(trainx, trainy, epochs=50, batch_size=256, verbose=False,\
                        callbacks=[early_stop],validation_data=(testx, testy),shuffle=True)
```

To train the LSTM model, I first normalize the features by the minmax scaler, which functions as the following formula. Standardization is not applied on the data because it led to higher symmetric mean absolute percentage error. Then split the features into training samples and test samples. The test samples are input into the model and the prediction result will be inverse transformed to the original data format. Finally the result will be evaluated by calculating the smape and mean square error between the prediction concentration values and real concentration values.

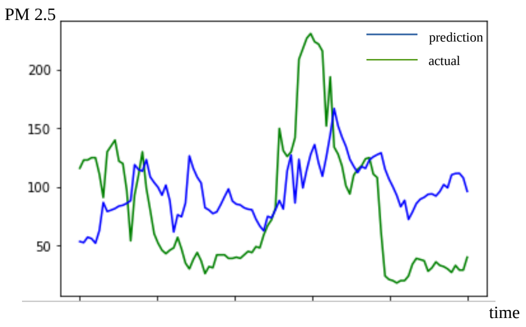
$$X' = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

Initially the weather feature was abandoned and data from January 1, 2017 to April 30, 2018 are input to train the model. One of the evaluation result is shown as following, where the prediction is the PM2.5 concentration at dongsi air quality station after 48 hours. The predicted time period is from April 26 to April 30, 2018. It can be seen that the real concentration tends to be more mutable and has high variability, while the prediction cannot capture it and seems to be more flats and changes slightly.



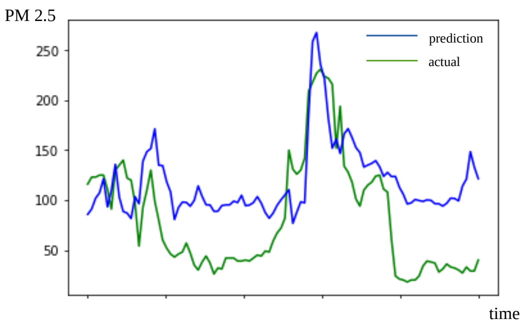
Mean squared error	4031.977
Smape	0.722501

To address this problem, weather features like humidity from other locations are input to train the model, because the weather conditions in the other places may also affect the air concentration. After utilizing features of 8 around grids from the grid weather files, the prediction is shown as following:



Mean squared error	4030.588
Smape	0.695656

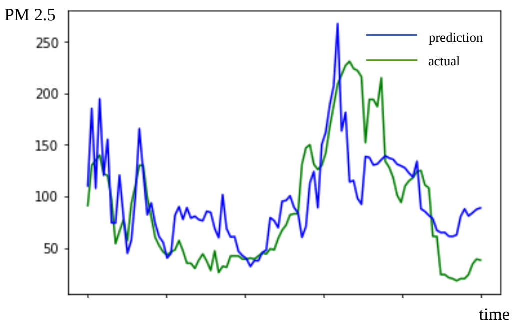
The graph illustrates the same problem as before. That may be because the data of the 8 grids has similar distribution as the station and cannot more information to the model. Therefore, I append more grid weather features from the outmost grids, and the result is illustrated below, because the



Mean squared error	3634.7328
Smape	0.643375

features from the outmost grids have higher variability than the grids around the station, the model has gained a better performance on both mean squared error and smape, as well as better ability of capturing sudden changes.

All models above are based on features without the categorical feature, weather. I also construct a new model utilizing data from April 1 to April 30, 2018 which contains the weather feature. I transform the weather features into numerical values by one hot encoding and evaluate the model by the same test samples. From the result below it can be found that the weather feature has huge impact on the air concentration and the performance has been improved significantly.



Mean squared error	1036.52098
Smape	0.3853830

Conclusion

In the project I predict the air concentration by LSTM neutral networks based on the past concentration information associated with meteorological parameters. I figure out the weather features in other places also have considerable impact on the air concentration and demonstrate it by utilizing data from grids which have large variance on longitude and latitude in model training and improving the model performance immediately.

There are still room for optimization. Most time is spent on feature engineering and there are not enough time for me to do model aggregation. In the future I may train tree model for prediction and aggregate it with neutral networks to improve the model performance.

Task assignment

Yuntao, Zhang: Write report, programming.