東海大學

資訊工程研究所

碩士論文

指導教授: 楊朝棟博士

運用機器學習分析於自動化預測空氣品質監測

Air Quality Monitoring and Analysis with Automatic Forecasting Using Machine Learning

研究生**:** 李靜芳

中華民國一零七年六月

東海大學碩士學位論文考試審定書

東海大學資訊工程學系 研究所

研究生 李 靜 芳 所提之論文

運用機器學習分析於自動化預測空氣品質監測

經本委員會審查，符合碩士學位論文標準。

學位考試委員會
召 集 人 許慶賢 簽章
委 員 呂芳懌
賴冠州
陳牧言

指 導 教 授 楊朝棟 簽章

中華民國 107 年 6 月 8 日

摘　要

時間序列預測問題是一種具有挑戰性的預測性建模案例研究，因為時間序列增加了輸入變量之間時間依賴性的捲積。然而，遞歸神經網絡（RNN）是一種功能強大的神經網絡，除了具有記憶性，且能夠處理時間序列相關的資料。因此在本文中，我們使用RNN進行空氣污染PM2.5的分析和其自動化預測。並且在實驗中，我們建立了一個基於RHadoop的分佈式計算環境，分析了空氣污染。除了將及時資料透過MySQL資料庫進行存取，也利用了Sqoop對HBase進行歷史數據快速存取。此外，我們也針對資料缺值補值進行討論及實驗，求出不影響預測精度或能提升預測精度的補值方法。並且在實驗中我們使用平均絕對誤差百分比（MAPE）值將PM2.5的短期預測精度進行量化，將MAPE控制再0.2至0.5區間。並且在不影響精度下，針對RNN各個參數進行實驗和校調，進一步開發RNN自動化訓練程式。最後，由於視覺化對於開發者的幫助以及使用者的影響，我們也使用R及Shiny來可視化RNN訓練結果，協助優化RNN訓練模組的參數，以利於開發者可以快速的分析資料資訊，並將預測為來PM2.5數值顯示於地圖上提供使用者參考。

關鍵字: 空汙分析，時間序列，機器學習，自動化預測，遞規神經網路

Abstract

The time series prediction problem is a challenging case study of predictive modeling because time series increase the time-dependent convolution between input variables. However, Recurrent Neural Network (RNN) is a powerful neural network that is not only memory but also capable of processing time series-related data. In this paper, we use RNN to analyze the air pollution PM2.5 and its automated prediction. Moreover, in the experiment, we established a distributed computing environment based on RHadoop and analyzed the air pollution. In addition to accessing timely data through the MySQL database, Sqoop is also used to access HBase historical data quickly. Also, we also discussed and experimented with a missing value of data to find a complementary method that does not affect prediction accuracy or enhance prediction accuracy. Moreover, in the experiment, we use the average absolute error percentage (MAPE) value to quantify the short-term prediction accuracy of PM2.5 and control the MAPE by 0.2 to 0.5 intervals. Moreover, without affecting the accuracy, experiment, and tune for each parameter of RNN, and further, develop RNN automated training program. Finally, because of the help of the visualization for developers and users, we also use R and Shiny to visualize the RNN training results and help optimize the parameters of the RNN training module so that developers can quickly analyze the information and The predicted PM2.5 value is displayed on the map for user reference.

Keywords: Air pollution, Time series, Machine Learning, Automatic Forecasting, RNN

致謝詞

對於分散式環境、機器學習、資料處理以及 R 語言的相關研究，一路下來面臨了相當多的挑戰。在這兩年的碩士生涯中，我很幸運能進入楊朝棟老師的研究生。楊老師每個禮拜都會固定與研究生討論進度，讓我們可以定出每個禮拜的進度並達成，讓我們碩士的研究進度更佳扎實。楊老師就像一個父親的角色，引導自己研究生方向，及使做的不盡完美也不過於苛責，對外也都稱讚自己研究生的長處，希望我們能夠真的成長。不僅讓我們學習如何做好一個研究，更讓我們學到如何與人的應對進退，以及如何與廠商、學長、學弟們進行團隊合作。在確定論文題目前我感到相當的迷惘，多虧了學長給予的意見以及指導，讓我能快速學習並發想論文方向。在進行論文時，也多虧學長、同學、學弟的幫助，讓我可以順利的進行實驗及撰寫。尤其特別感謝我的同學王元廷與陳子揚，他們陪伴我研究所的生涯，互相幫助學習，一起參與比賽以及研討會。我也特別感謝我的父母對於我求學階段的支持與鼓勵，讓我可以無後顧之憂的完成碩士學業。還有感謝特地抽空前來參加的口試委員們: 許慶賢教授、呂芳懌教授、賴冠州教授、以及陳牧言教授，感謝各位教授所給予的寶貴建議，因為有您們的幫助，讓我看到自己論文的優缺點並能夠加以改進，讓我的論文能夠更加完善。最後，感謝所有一路上幫助過我的人，因為有你們的付出才能成就今天的我，謝謝你們。

東海大學資訊工程學系高效能計算實驗室李靜芳 107 年 06 月

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Time series prediction problems [?] are a challenging type of predictive modelling case study as time series adds the involution of a temporal dependence among the input variables. Recurrent Neural Network (RNN) [?] [?] are able to handle time sequence problem because they can retain state from one iteration to the next by using their own output as input for the next step. Their ability to form a directed cycle have been proving that RNN are a powerful predictive engines specifically for time series data. While air pollution dataset is a time series data that have a natural temporal ordering and continuous data. Therefore, RNN are useful in time series modelling and prediction for air pollution dataset [?] [?].

In this study, we aim to implement a Recurrent Neural Networks for analysis and automated air pollution forecasting. Specific goals are:

1. To deploy a distributed computing environment based on RHadoop,
2. To analyze air pollution and presented visualization using HBase from historical data.
3. To analyze the short-term prediction of PM2.5 and measured the prediction accuracy based on mean absolute percentage error (MAPE) value.

4. To utilize shiny to visualize the training result for optimizing parameters of RNN training module.

1.1.1 The Impact of Air Pollution on Health

The level of pollution endangering the health of the body is quite extensive. Because the pore size of these air pollutants is very small, as the breath enters the lungs, it is easy to cause inflammation of the lungs first. When these inflammatory factors in the body are transported to the body with blood, May become blood vessels, systemic inflammation.

National Institute of Environmental Medicine of the National Guardian Institute pointed out that airborne air pollution contains fine suspended particulates (PM2.5), nitrogen oxides, cerium oxides, carbon monoxide, and ozone. Overall, Taiwan’ s ranking of international emissions of PM2.5 is medium exposure, which is still higher than in North America, some European countries, and Australia. Many studies have confirmed the harm of PM2.5 to human health.

1.1.2 The Importance of Big Data in Air Pollution

Inhalation of too much-contaminated air or particulate contaminants by the human body will deprive the respiratory organs of normal defensive and removal functions and endanger health. Many pollutants in the atmosphere coexist, so the relationship between air pollutants and human diseases is difficult to confirm clearly. Diseases such as lung cancer, chronic bronchitis, and pulmonary edema are considered to be related to air pollution.

For example, sulfur oxides, nitrogen oxides cause acid rain, or carbon dioxide causes the reduction of heat dissipation in the atmosphere, which causes the surface temperature of the earth to rise, creating a greenhouse effect and so on, all of which have an impact on the climate. Also, particles in the air will scatter sunlight, reducing the energy of sunlight reaching the earth and reducing the temperature of the earth. This impact on the climate will create a chain effect because the climate itself also affects many levels. From this point of view, the scope of influence mentioned earlier has been almost covered; it can also be said that air pollution has caused an impact on the entire ecosystem.

1.1.3 Time Series Data Analysis

An extension of historical data, also known as historical extension prediction. It is based on the development process and regularity of the socio-economic phenomena that can be reflected by the time series and the method of extrapolation and prediction of its development trend. Time series, also known as time series, historical plurals, or dynamic sequences. It is to sort the numerical values of certain statistical indicators in order of time into the sequence formed. Time series forecasting method is to compile and analyze time series and analogize or extend the development process, direction, and trend reflected by time series, to predict the level that may be reached in the next period or the following years. Its contents include: collecting and sorting historical data of certain social phenomena; examining and identifying these data and arranging them into series; analyzing the time series, looking for the law of the social phenomenon changing with time, and drawing a certain pattern; This model predicts the future of this social phenomenon.

1.2 Thesis Goal and Contributions

As the impact of air pollution on the human body is visible, this paper uses the prediction analysis of PM2.5 to observe the trend of PM2.5 and provides further analysis of other studies. It hopes to help improve the air pollution problem.

Therefore, this paper proposes an integrated system architecture that uses R for time series analysis, starting from the initial data capture, data preprocessing, and the use of mathematical modules to patch missing data before these analyses. Then, through machine learning RNN, the parameters of data training prediction are corrected, the trend of the parameters is analyzed, and the RNN training process and forecasting results are visualized. Finally, assisting the experiment with visualization to obtain better training modules and further analysis.

Integrated R-distributed computing system in the data reading part, HBase and MySQL are combined through Sqoop, and a large amount of geographic information can be read directly through RMySQL. In the repair missing data and remove outliers section, the steps meaningful values and select data parameters.

Due to the need to be able to deal with a large amount of data on the read and write platforms, it is an indispensable part of data analysis. Moreover, in the process of Data Transformation and Data Mining in R in the single machines, the performance is greatly limited by the memory. The R- distributed computing system uses Rhadoop's R packages. This package is the API interface to the Hadoop distributed environments. Rhadoop breaks through the limitations of R in the single machines, and effectively improves R's ability to analyze large amounts of data and speed, and allows R's analysis environment to connect multiple single machines to the cluster server.

1.3 Thesis Organization

The chapter 2 explains the relevant knowledge of this article and the application of R language packages such as Rhadoop, HBase, Sqoop, and visualized shiny server. Also, it will also analyze the air pollution of this system. Moreover, we explain the relevant mathematical models in detail, including RNN's training and predictive analysis of time series and using MAPE as the quantification of RNN prediction accuracy. In the chapter 3, we explain how we have integrated the R language packages into Rhadoop and the shiny visual server into a complete system, and implement the HDFS decentralized storage and use of Hadoop environment in the reading and writing of data. Sqoop makes MySQL, And Hbase can access each other. It will introduce the importance of Lag-Time for RNN training module and how it affects the accuracy of RNN. As well as introducing the relationship between Rstudio and Shiny Server, and briefly mentioning the importance of visualization. At the end of this chapter, a brief explanation of the set of automated training flowcharts designed for the experiment is provided. The chapter 4 includes experiments from the system design, data capture, data access to the database, data preprocessing, data missing, RNN training, and predictive visualization. The focus of this chapter is on the patching of data and RNN's experimental analysis of PM2.5 predictions. Finally, this section of the chapter 5 summarizes the system and experiment.

Chapter 2

Background Review and Related Work

In this section, we review some background knowledges for later use of system design and implementation.

2.1 Background Review

2.1.1 Machine learning

Machine learning involves recognizing the operating mode through a sample training machine, rather than programming with specific rules. These samples can be found in the data. In other words, machine learning is a kind of weak artificial intelligence . It gets complex functions from data to learn to create an algorithm and use it to make predictions.

Although the predictive power of the model is important, it is not all. Some algorithms are easy to explain, and some algorithms are very good at dealing with issues such as noise and missing data. So the algorithm for picking machine learning is very loyal. No algorithm can solve all problems, but there are many good algorithms.

Machine learning has been widely used in data mining, computer vision, natural language processing, biometrics, search engines, medical diagnostics, detection of credit card fraud, stock market analysis, DNA sequencing, voice and handwriting recognition, strategy games, and robotics.

2.1.2 Recurrent Neural Networks

RNN is a tier of artificial neural network (ANN) [?] where connections between units form a directed graph along a sequence. In a traditional neural network [?]. all inputs and outputs are independent of each other. This method is not suitable for some cases spesifically in time series dataset forecasting. For example, if we want to predict the next word in a sentence we better know which words came before it. In this case, RNN are able to solve this problem because their output being dependent from previous computation. Moreover, RNN have a short-term memory which captures sequence data about what has been calculated so far.

Figure 2.1 is the concept of RNN module operation, including adjustable parameters. In order to be able to manually retrain the module, we designed a process of automatic training.

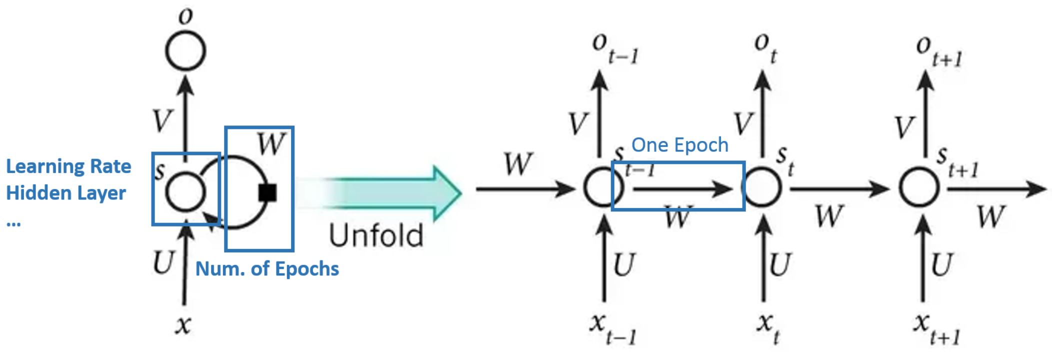


FIGURE 2.1: How RNN Works

It can be seen that the final result O (t + 1) of the network at time (t + 1) is the result of the accumulation and computing between input and all history data at that moment. The above structure expands RNN into a complete network.

For example, if you need to process a 5-level sequence, expanding to 5 levels is necessary.

- ‘x’ means the network input at a moment;
- ‘s’ is hidden status, representing the ”memory” of the network;
- ‘o’ represent the output.

The hidden state ‘s’ can be thought of as the ”memory” of the network, which captures information all the times before. The output ‘o’ depends only on the memory at the current moment, and it is very difficult for ‘s’ to capture long-lived messages due to long-term dependencies actually. Therefore, the parameters (U, V and W) in each step of RNN are the same, which reduces the cost of learning a lot. Depending on the actual task, you don’t need to input/output something for each step. The algorithm is shown in Figure 1.

Algorithm 1 Recurrent Neural Network forecasting Algorithm

```
Set $rnn_units$ and $optimizer$ to define RNN Network ($R$);
Normalize the dataset ($D_i$) into values from 0 to 1 ;
Select training $windowsize(tw)$ and organize $D_i$accordingly;
for $number\_of\_epochs$ and $batch\_size$ do
    Train the $Network(R)$ ;
end for
Run Predictions using $R$;
Calculate the $loss\_function$;
```

In the preprocessing step, the data is normalized to a range of 0 to 1. The data normalization formula is as follows:

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

Moreover, set RNN parameter input the definition of the completed RNN module, and it is reconstructed according to the training *windowsize*(*tw*). The training window is defined as a set of patterns for predicting the next pattern. Then repeat the training through the defined RNN module, using *loss_function* to observe the accuracy of the prediction.

2.1.3 Visualizations R Using the Shiny Server

Visualization of data means visually presenting data [?]. Effective graphs can simplify complicated data into easily absorbed content. Through graphical means, we can more easily discern Patterns, Trends, and Correlations. The earliest data was traced back to the 17th century when people plotted maps; in the early 18th century, humans invented the Pie Chart. Visualization of data appears in our lives all the time, magazine articles, news media, academic reports, public transportation instructions, and more. Infographic Information chart is another term we often hear. It has the same purpose as data visualization —using graphical methods to simplify complex information. However, Infographic is more subjective and qualitative information, expressing the author's viewpoints and stories. More emphasis on visual presentation requires deeper design skills.

Shiny [?], a suite developed by R's new startup team, R Studio, is now playing an important role in prototype production in many companies' internal research projects. Moreover, Shiny Server [?] is one of a Web development framework in R language that contains many front-end business intelligence (BI)-packages, like ShinyBI, ShinyJS, Shinydashboard, etc. Furthermore, Shiny combined with data visualization packages such as ggplot2, ggmap [?], leaflet, REmap, etc. in the R language, which can be presented as an interactive web and, moreover, provides developers analysis and forecasting easier. Shiny's operation is mainly divided into front-end ui.R and back-end server.R. These two parts ui.R and server.R communicate by output $ tag and input $ tag. The ui.R web interface is similar to HTML in syntax command.

2.2 Related Work

In the training of prediction modules with time series, most of them focus on the improvement of accuracy and the influence of data correlation on training results. The main references are as follows: D. Lee et.al. [?] predicted the pro-environmental consumption index based on Google search query data, using a recurrent neural network (RNN) model. Also, they compared the prediction accuracy of the RNN model with that of the ordinary least square and artificial neural network models. Stefan Balluff and Stefan Krauter [?] presented current results of wind speed forecasts using recurrent neural networks (RNN) and the gradient descent method plus a backpropagation learning algorithm. Mohamed Akram Zaytar and Chaker El Amrani [?] proposed a deep neural network architecture and use it in time series weather prediction. Camelo, et al. Paper [?] predicted time series applied in wind generation based on the combination of time series models with artificial neural networks. Rahman A. et al. In [?] predicted electricity consumption for commercial and residential buildings using deep recurrent neural networks. Myttenaere, et al. Many papers compare RNN to LSTM. The conclusion is that no one module is the best, but only suitable for this data. Moreover, in our paper, we also conducted the prediction of PM2.5 accuracy by RNN and LSTM.

Machine learning is also a challenge for training from time-series data. Like the paper [?] [?], they proved although the implementation of National Health Insurance, it has accumulated enough data to establish a time series forecasting model. In addition to the previous medical expenses, in order to understand the impact of other exogenous variables on medical expenses, this paper combines time series. Models and causal models to predict total health care expenditure for National Health Insurance.

After deciding which chemical factor to use for the prediction of PM2.5, in addition to the correlation of each air quality value to the external, we also refer to the chemical-related papers for corroboration. In the article [?], they analyzed the data of air quality and found that the relationship between O3 and AQI was negatively correlated. Also, AQI is a weighted calculation of various air quality values. PM2.5 is also a part of it. For better accuracy, we do not predict a value that has already been calculated but instead, forecast for PM2.5. That is why our paper chose to use O3 to predict PM2.5.

The quantification of prediction accuracy is also very important. For the selection of quantization modules, we refer to several papers. In the article [?], they proved the existence of an optimal MAPE model and showed the universal consistency of Empirical Risk Minimization based on the MAPE. Our work proposed the implementation of RNN in analyzing and automated predicting of air pollution. Also, in the paper [?], they also use the MAPE value to quantify the accuracy of the data. In the article [?], due to the inherent nature of memory design, it is difficult to effectively and reliably analyze air pollution data in standalone environments. Moreover, they use sensors called EdiGreen AirBox to collect data on air pollution in Taichung. Finally, the experimental results show that the accuracy of the short-term forecast results of PM 2.5 is analyzed using the ARIMA model. Also, verifying the prediction accuracy of the MAPE method is also given in the experimental results. Therefore, this paper decided to use MAPE to quantify accuracy. Although MAPE can't see the data trend, it proves that the MAPE still has its use value through the experimental chart.

Mathematical analysis and other features, through the R language installation kit, to expand the R language function [?], with the array and matrix calculation capabilities, display of the drawing tools and simple and easy programming language. For data screening, repeated calculations, import/export of data, and development of custom program functions [?]. Compared with other programming languages, the biggest advantage is the drawing function, which includes a variety of image rendering effects, visualized graphics after analysis, and a good level of print quality. The number of Packages supporting R language can be used as a tool for analyzing R language.

Chapter 3

System Design and Implementation

3.1 System Architecture

We use python and R language to store the air pollution value of open data and geographic information of TGOS in the server database, as shown in Fig. 3.1. Then we apply R language for analysis and prediction. Next, we use the plot in the shiny server, ggplot2, Plotly packages to visualize the data so that we can conduct preliminary analysis of the experiment for subsequent prediction module optimization. Finally, we utilize leaflet, ggmap packages and TGOS geographic information to present the data to the user in an interactive map.

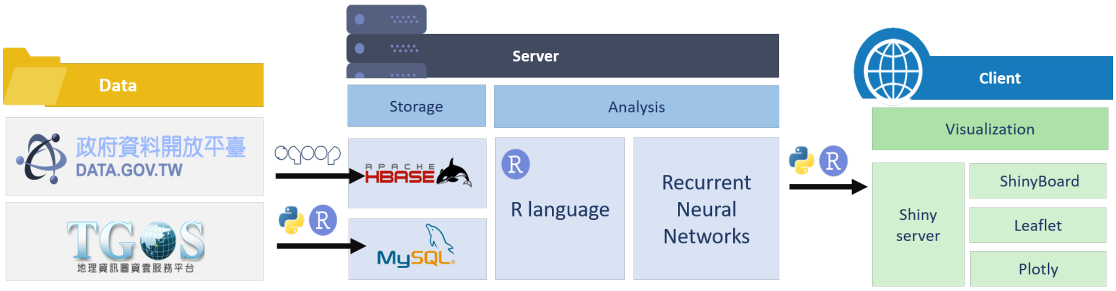


FIGURE 3.1: System Analysis Process

We used one master and two slaves for Hadoop cluster as the experimental environment and used RHadoop environment. We also built HBase on the master and used Sqoop as the bidirectional data transfer for MySQL to HBase [?] [?], as a Figure 3.2. MySQL is used to store the time series data and run it using R-base [?]. For fast data access we used RMySQL package. Since HBase has exclusive access to big data quickly, we use historical data to access it using HBase. Finally, we present the data to Shiny-Server via R-base.

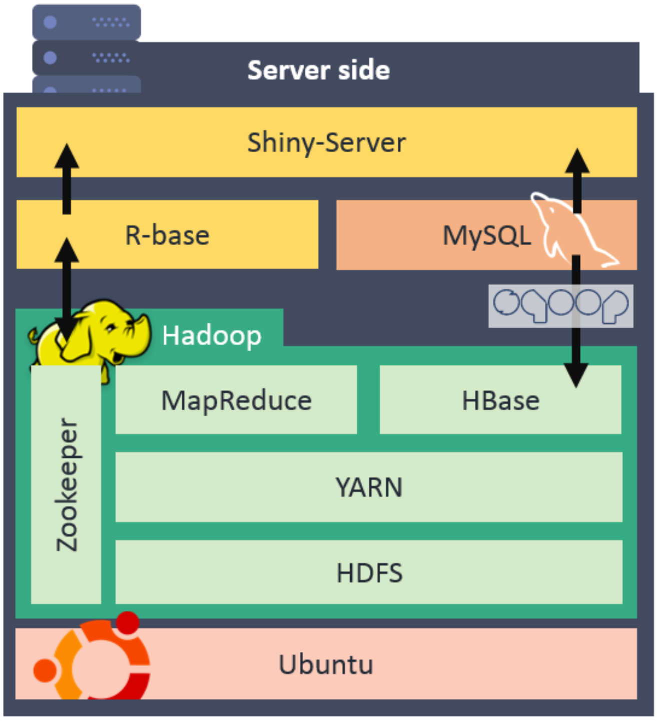


FIGURE 3.2: Hadoop Cluster's Master Architecture

3.1.1 Rhadoop

R is a popular and influential analytical language. Previously, the stand-alone R language computing power is limited by memory size, which makes analysis environment of R hardly to implement in Big Data. By the time, the advanced Rhadoop architecture enables R's data analytic [?] environment to have the ability in dealing with big data using Hadoop. Rhadoop [?] can be seen as Hadoop's environment in R interface, MapReduce [?], HDFS [?], and HBase [?]. The packages of rmr2, rhdfs, and rhbase are developed for three environments, MapReduce, HDFS, and HBase.

3.1.2 HBase

HBase is a distributed, column-oriented, open-source database that is derived from the Google paper "Bigtable: A Distributed Storage System for Structured Data"[?] by Fay Chang. Just as Bigtable leverages the distributed data storage provided by the Google File System (File System) [?], HBase provides similar Bigtable capabilities on top of Hadoop. HBase is a subproject of Apache's Hadoop project. HBase is different from the general relational database. It is a database suitable for unstructured data storage. So we chose to use HBase for historical data access.

3.1.3 Sqoop

Apache Sqoop (SQL to Hadoop) [?] is designed to support bulk import of data from structured data stores to HDFS, such as relational databases, enterprise data repositories, and NoSQL systems. Sqoop is based on a connector architecture that supports plug-ins to provide connectivity to new external systems. Sqoop Connector works with a variety of popular relational databases, including MySQL, PostgreSQL, Oracle, SQL Server, and DB2. Each of these connectors knows how to interact with its associated database management system. There is also a Universal JDBC Connector for connecting to any database that supports the Java JDBC protocol. Also, Sqoop offers optimized MySQL and PostgreSQL connectors using database-specific APIs to perform bulk transfers efficiently.

3.1.4 Mean Absolute Percentage Error

Mean Absolute Percentage Error(MAPE) [?] used as an evaluation indicator for the prediction model. Because MAPE is a relative value, it is not affected by

Table 3.1: The Evaluation Criteria of MAPE

MAPE	Instructions
<0.1	Highly accurate prediction
0.1-0.2	Excellent forecast
0.2-0.5	Reasonable forecast
>0.5	Inaccurate prediction

the unit and size of the measured value and estimated value, and can objectively obtain the difference between the estimated value and the assessed value, as shown in the formula below:

$$MAPE = \frac{1}{M} \sum_{k=1}^{M} \left| \frac{x(k) - x'(k)}{x(k)} \right|$$

among them,

$$x(k) - x'(k) = \varepsilon_k$$

Actual value:

$$x(k)$$

Estimated value:

$$x'(k)$$

Number of samples:

$$M$$

Error per vehicle:

$$\varepsilon_k$$

The closer the MAPE value is to 0, the better the estimation effect, as shown in Table 3.1. Also, Lewis, the scholar, believes that MAPE is the most effective evaluation indicator and sets the relevant evaluation criteria.

In summary, the operations manager must determine the relative importance of historical performance on responsiveness and whether MAD, MSE or MAPE?? should be used to measure historical performance.

- MAD is the easiest to calculate, but it has the same weight for all errors.
- The MSE weight is based on its square value, but it also has a large error so that it will cause more problems.
- When the error needs to be measured in a relative point of view, MAPE is used.

3.2 Lag-time in RNN for Forecasting

There are two main ways to make predictions through Lag-Time. The first one is to do the Lag-Time displacement beforehand and then perform the RNN training, as a Figure 3.3. The second is to perform the RNN training first and then do the Lag-Time displacement. We choose the first in this experiment. This method uses O3 to predict PM2.5, and the two trends are negatively correlated, as a Figure 3.4. Therefore, if the Lag-Time shift is performed first, the positive correlation of data increases and the accuracy of RNN prediction [?] will be significantly improved.

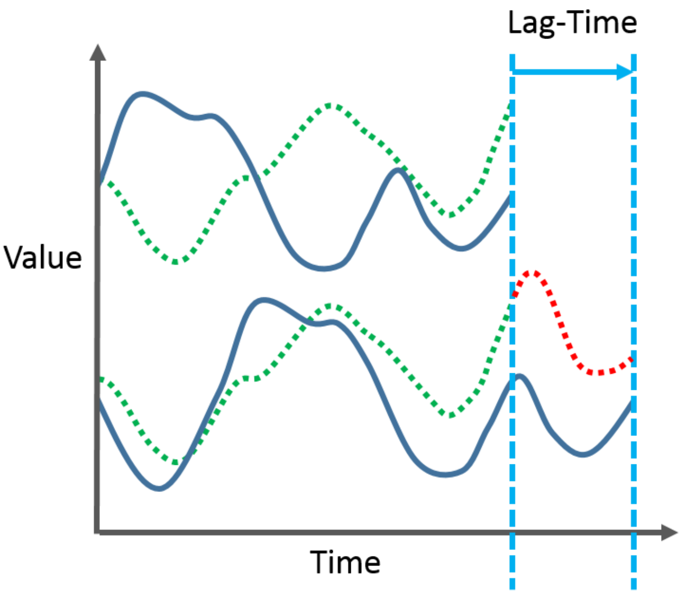


FIGURE 3.3: Using Lag-Time to RNN Training

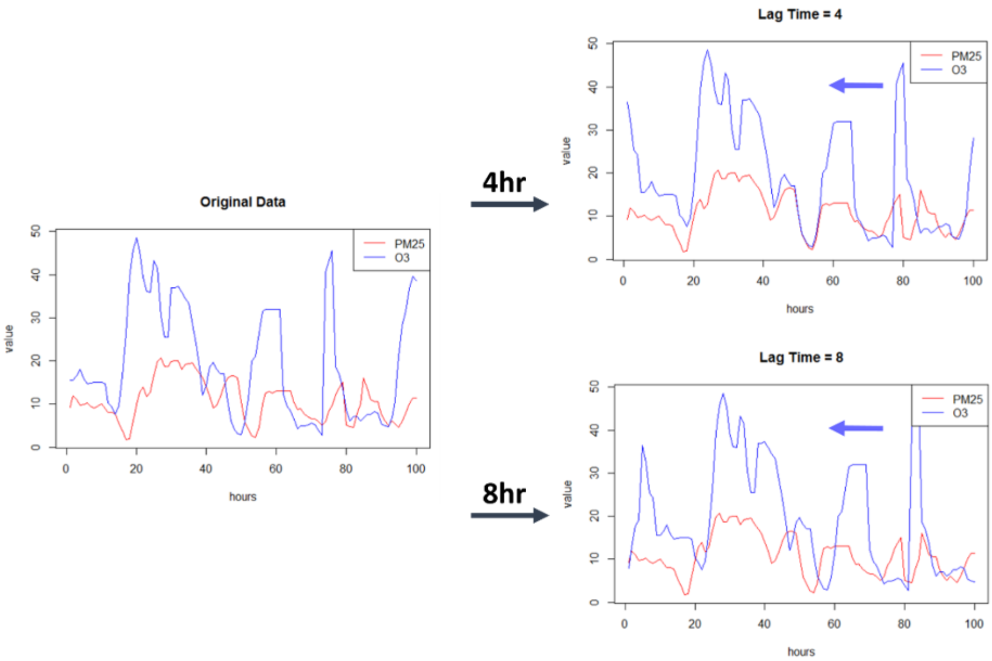


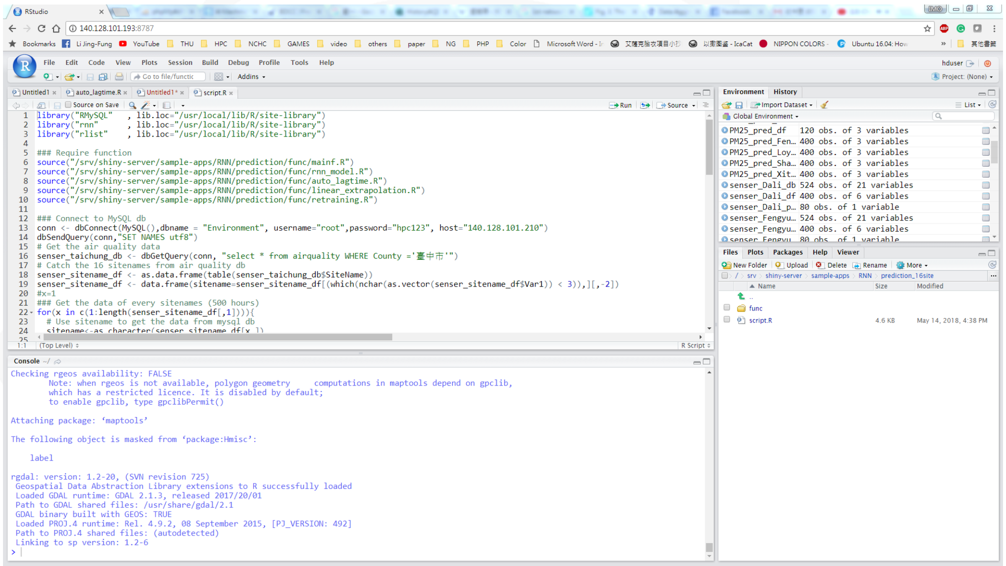
Figure 3.4: Different Lag-Time Correlations for PM2.5 and O3

After observing the air pollution data, we observed that Lag-Time is not a fixed value, so in this work, we designed it to detect the best Lag-Time automatically. The programming concept is as follows:

1. Input PM25's data frame and O3's data frame, which contains air pollution data and time data.
2. Define the maximum and minimum values of the Lag-Time period.
3. Push back the Lag-Time period at the current time, and grab the highest PM2.5 time in PM25's data frame.
4. The receiver pushes back the highest time point of PM2.5 in PM25's data frame back to a Lag-Time period and grabs the highest time point of O3 in O3's data frame.
5. The highest time point of pm2.5 is subtracted from the highest time point of O3 and Lag-Time is available.

3.3 Rstudio and Shiny Server

In this experiment, we use Rstudio and Shiny Server for program writing and visual interface design. By the official website to provide the steps to install the Rstudio server environment, and in the IP with http: // <server-address>: 8787 /login to operate Rstudio, as a Figure 3.5.



```
library("RMySQL"    , lib.loc="/usr/local/lib/R/site-library")
library("rnn"       , lib.loc="/usr/local/lib/R/site-library")
library("rlist"     , lib.loc="/usr/local/lib/R/site-library")

### Require function
source("/srv/shiny-server/sample-apps/RNN/prediction/func/mainf.R")
source("/srv/shiny-server/sample-apps/RNN/prediction/func/rnn_model.R")
source("/srv/shiny-server/sample-apps/RNN/prediction/func/auto_lagtime.R")
source("/srv/shiny-server/sample-apps/RNN/prediction/func/linear_extrapolation.R")
source("/srv/shiny-server/sample-apps/RNN/prediction/func/retraining.R")

### Connect to MySQL db
conn <- dbConnect(MySQL(),dbname = "Environment", username="root",password="hpc123", host="140.128.101.210")
dbSendQuery(conn,"SET NAMES utf8")
# Get the air quality data
senser_taichung_db <- dbGetQuery(conn, "select * from airquality WHERE County ='臺中市'")
# Catch the 16 sitenames from air quality db
senser_sitename_df <- as.data.frame(table(senser_taichung_db$SiteName))
senser_sitename_df <- data.frame(sitename=senser_sitename_df[(which(nchar(as.vector(senser_sitename_df$Var1)) < 3)),][,-2])
#x=1
### Get the data of every sitenames (500 hours)
for(x in c(1:length(senser_sitename_df[,1]))){
  # Use sitename to get the data from mysql db
  sitename<-as.character(senser_sitename_df[x,])
```

```
PM25_pred_df     120 obs. of 3 variables
PM25_pred_Fen…   400 obs. of 3 variables
PM25_pred_Loy…   400 obs. of 3 variables
PM25_pred_Sha…   400 obs. of 3 variables
PM25_pred_Xit…   400 obs. of 3 variables
senser_Dali_db   524 obs. of 21 variables
senser_Dali_df   400 obs. of 6 variables
senser_Dali_p…   80 obs. of 1 variable
senser_Fengyu…   524 obs. of 21 variables
senser_Fengyu…   400 obs. of 6 variables
senser_Fengyu…   80 obs. of 1 variable
```

```
/ > srv > shiny-server > sample-apps > RNN > prediction_16site
..
func
script.R                 4.6 KB     May 14, 2018, 4:38 PM
```

```
Checking rgeos availability: FALSE
        Note: when rgeos is not available, polygon geometry     computations in maptools depend on gpclib,
        which has a restricted licence. It is disabled by default;
        to enable gpclib, type gpclibPermit()

Attaching package: ‘maptools’

The following object is masked from ‘package:Hmisc’:

    label

rgdal: version: 1.2-20, (SVN revision 725)
 Geospatial Data Abstraction Library extensions to R successfully loaded
 Loaded GDAL runtime: GDAL 2.1.3, released 2017/20/01
 Path to GDAL shared files: /usr/share/gdal/2.1
 GDAL binary built with GEOS: TRUE
 Loaded PROJ.4 runtime: Rel. 4.9.2, 08 September 2015, [PJ_VERSION: 492]
 Path to PROJ.4 shared files: (autodetected)
 Linking to sp version: 1.2-6
>
```

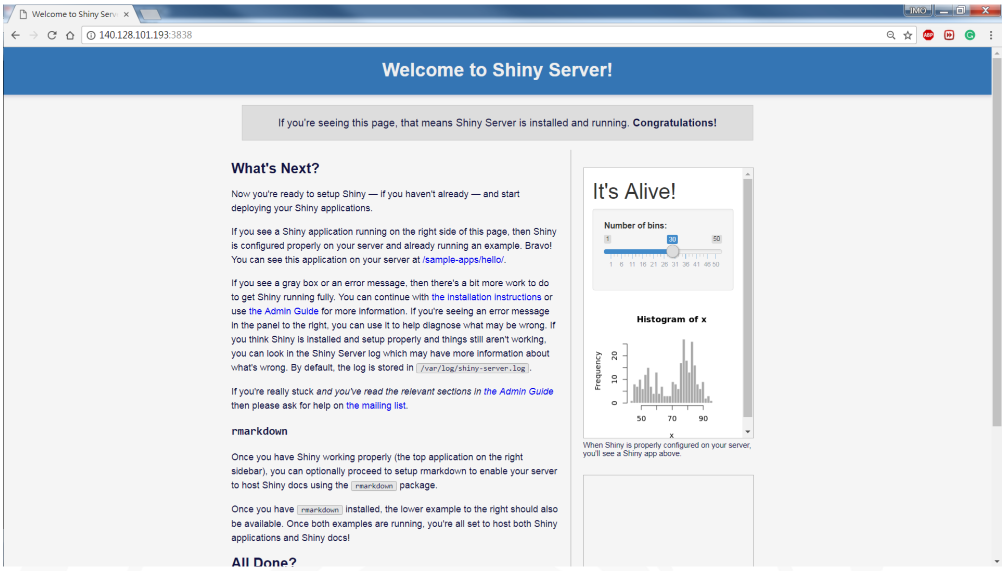
FIGURE 3.5: The Rstudio server web image interface

The shiny server and Rstudio server installation method are very similar, according to Rstudio official website to provide the steps to install the Shiny server environment, and use http://<server-address>:3838/ IP login. When us input IP, We can see the Shiny server’ s initial page, as a Figure 3.6.

If ip is changed to http://<server-address>:3838/sample-apps/, it will go to shiny server sample-apps directory. Developers can build new Shiny web directories in this directory, as a Figure 3.6.

3.4 RNN Model Optimization

When re-training the RNN model, it is necessary for the developer to adjust the parameters himself to find a suitable model. However, when the accuracy is reduced, the parameters and the training model must be recalibrated. It is difficult



Welcome to Shiny Server!

If you're seeing this page, that means Shiny Server is installed and running. **Congratulations!**

What's Next?

Now you're ready to setup Shiny — if you haven't already — and start deploying your Shiny applications.

If you see a Shiny application running on the right side of this page, then Shiny is configured properly on your server and already running an example. Bravo! You can see this application on your server at /sample-apps/hello/.

If you see a gray box or an error message, then there's a bit more work to do to get Shiny running fully. You can continue with the installation instructions or use the Admin Guide for more information. If you're seeing an error message in the panel to the right, you can use it to help diagnose what may be wrong. If you think Shiny is installed and setup properly and things still aren't working, you can look in the Shiny Server log which may have more information about what's wrong. By default, the log is stored in `/var/log/shiny-server.log`.

If you're really stuck *and you've read the relevant sections in the Admin Guide* then please ask for help on the mailing list.

`rmarkdown`

Once you have Shiny working properly (the top application on the right sidebar), you can optionally proceed to setup rmarkdown to enable your server to host Shiny docs using the `rmarkdown` package.

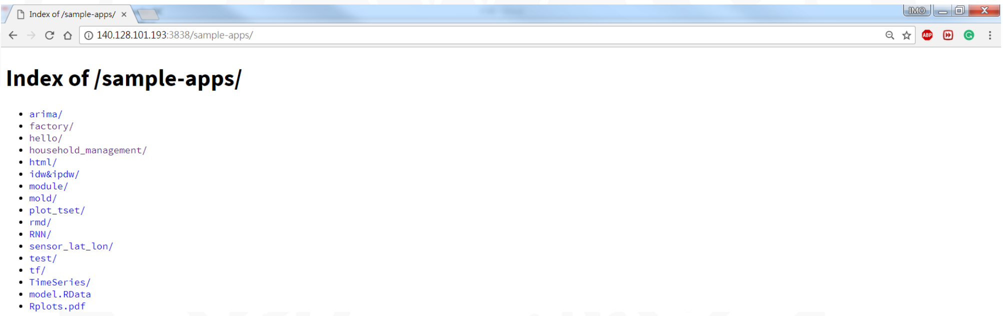
Once you have `rmarkdown` installed, the lower example to the right should also be available. Once both examples are running, you're all set to host both Shiny applications and Shiny docs!

All Done?

It's Alive!

When Shiny is properly configured on your server, you'll see a Shiny app above.

FIGURE 3.6: The Shiny server start page



Index of /sample-apps/

- arima/
- factory/
- hello/
- household_management/
- html/
- idw&ipdw/
- module/
- mold/
- plot_tset/
- rmd/
- RNN/
- sensor_lat_lon/
- test/
- tf/
- TimeSeries/
- model.RData
- Rplots.pdf

FIGURE 3.7: The sample-apps directory in the shiny server

to find a significant change in the weather now. A single good model is used for long-term prediction. In this experiment, we demonstrated not only automatic Lag-time detected, but also some of the parameters in the RNN are automatically calibrated. The flowchart is shown in Fig. 3.8. How to select which parameters are suitable for automatic detection and calibration, is also the focus of this experiment. The adjustment will be mentioned in the fourth chapter of the experiment.

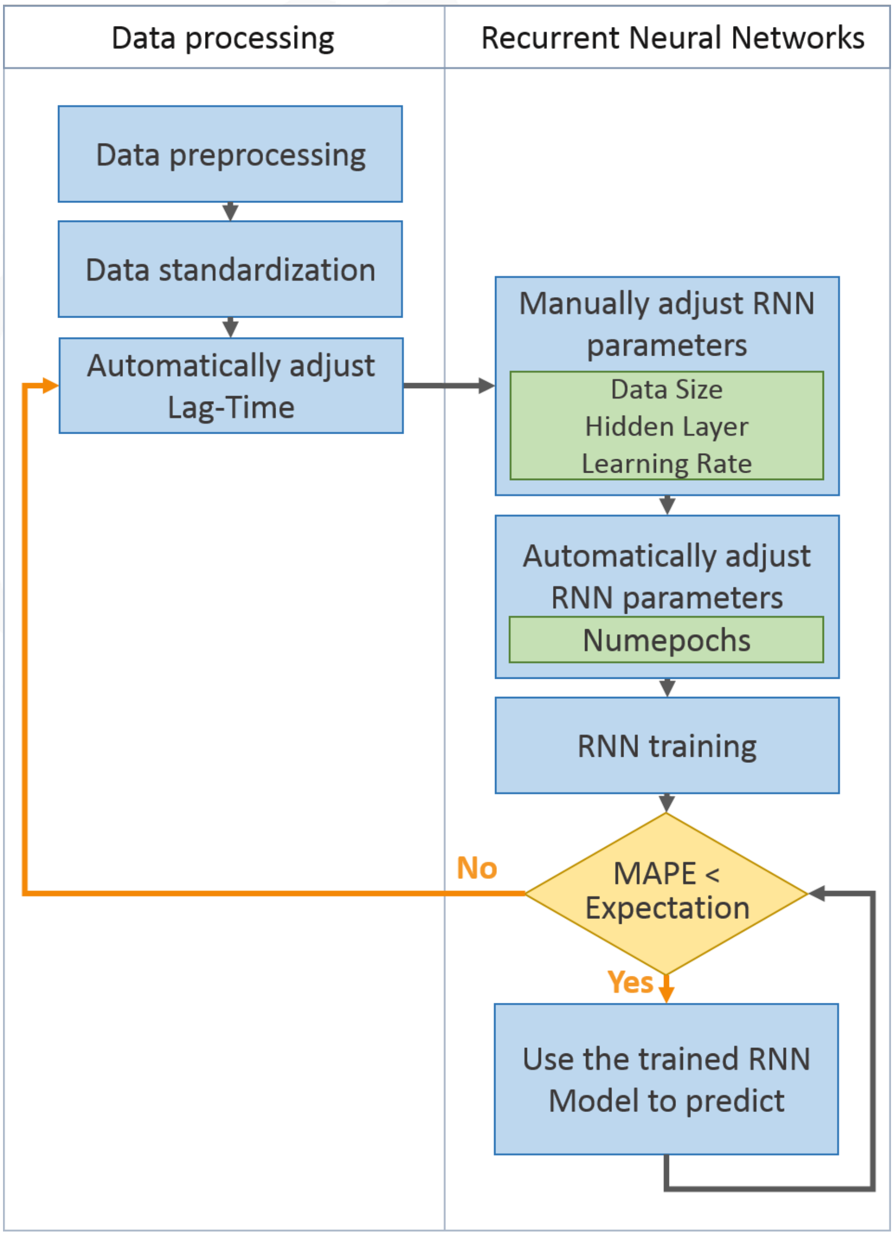

Data processing
Recurrent Neural Networks
Data preprocessing
Data standardization
Automatically adjust Lag-Time
Manually adjust RNN parameters
Data Size
Hidden Layer
Learning Rate
Automatically adjust RNN parameters
Numepochs
RNN training
MAPE < Expectation
No
Yes
Use the trained RNN Model to predict

Figure 3.8: RNN Training Process

Chapter 4

Experimental Results

4.1 Experimental Environment

This experiment uses three physical machines, a Master, and two slaves, as a Hadoop cluster, its specifications are shown in Table 4.1. Moreover, each version number is shown in Table 4.2. Also, noteworthy is the Sqoop and HBase version numbers compatibility in the built environment.

Table 4.1: Distributed Computing Environment

	CPU	RAM	Disk
Master	Intel(R) Xeon(R) CPU E5645 @ 2.4GHz * 2	64G	2T * 2
Node1	Intel(R) Xeon(R) CPU E5645 @ 2.4GHz * 2	64G	2T * 2
Node2	Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz	16G	1T

Table 4.2: Software Specification

Version
Ubuntu 16.04.3 LTS
r-base-1.0.136
shiny-server-1.5.1.834
hadoop-2.8.1
hbase-0.95.2-cdh5.0.0
sqoop-1.4.6
zookeeper-3.4.5

4.2 Distributed Computing

To achieve automatic forecasting and retrain when predictions are inaccurate, we use a decentralized architecture to reduce the overall operating time. Moreover, through Hadoop-supported HBase access to historical data, to prevent the future encounter the problem of excessive data capacity.

4.2.1 Multi-core for RNN Training

Because R does not support multi-core computing, we use the parallel package to use the lapply function to rewrite the program to use multi-core operations. Experiments are performed using a 4-core machine. The results are shown in Figure 4.1. The performance of programs optimized in the single core is poor, but performance is better than that of optimized programs at almost the core, and it is the best performance at 4 cores.

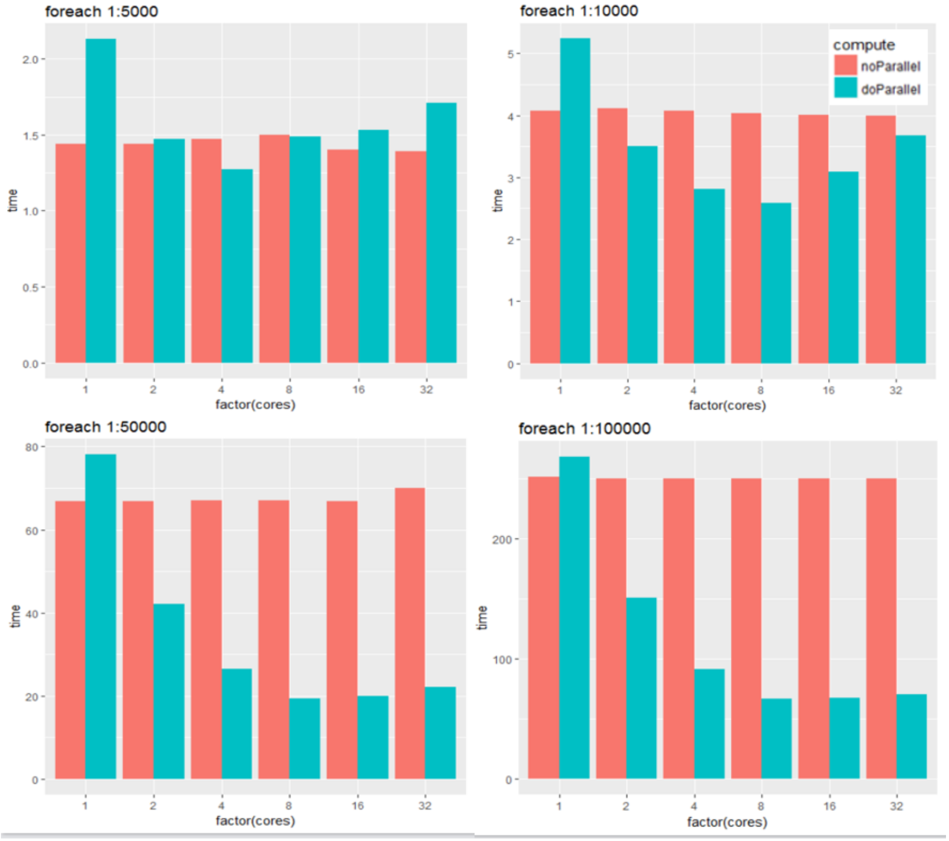


FIGURE 4.1: Difference times of Loop

Then we rewrite the RNN module so that it can use multi-core resource operations, but limited by the package; only the loop program can be optimized. However, since the RNN core technology is not a combination of loops, it can be seen that the performance has not greatly improved. The results are shown in Figure 4.2 and 4.3. R is indeed a big disadvantage in the allocation of computing resources.

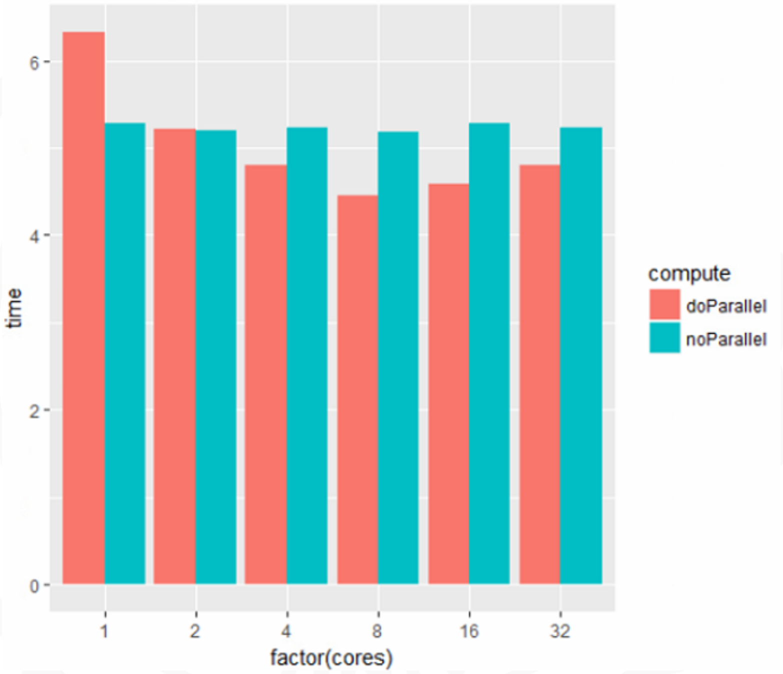


FIGURE 4.2: Number of Epochs set 2000

4.3 Data Preprocessing

4.3.1 Repair Missing Data

In the automatic Lag-time, the problem of missing data is mainly encountered, as shown in Figure 4.4. If there are missing values in the data, the prediction must be a loss of value. Therefore, we must repair the missing data to achieve the use automatic Lag-Time for RNN prediction.

To achieve the automatic Lag-time, we first need to fill in the missing data. In this work, we compare the accuracy of four different compensation methods for

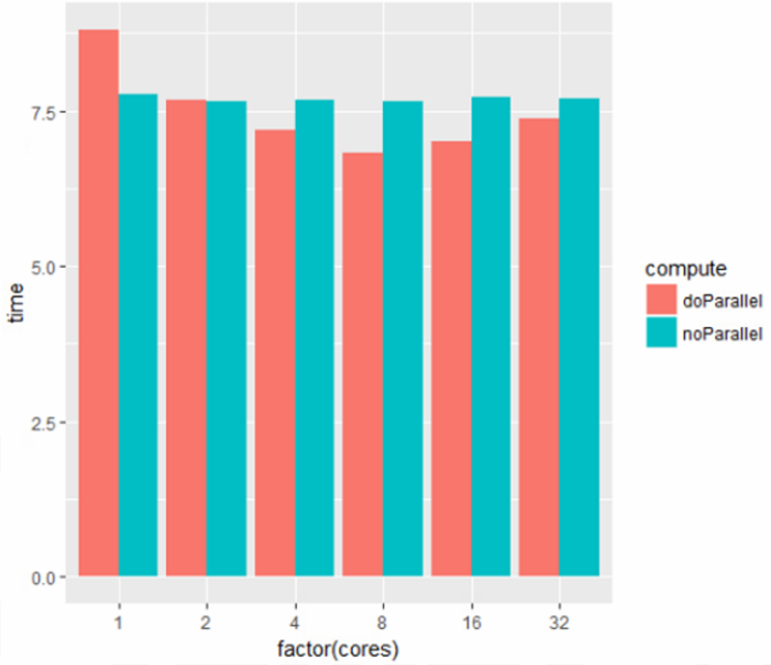


Figure 4.3: Number of Epochs set 4000

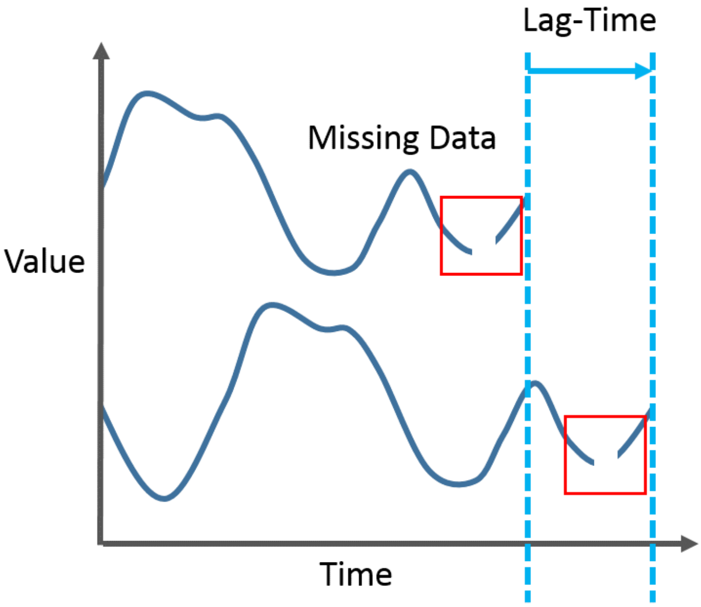


Figure 4.4: Impact of Missing Data on Lag-Time Prediction

missing data, including Zero, Mean, Random and Linear Extrapolation. Before data being repaired, first, we analyze the missing data. As can be seen, the graph 4.5 is visualized using VIM package nested marginplot() function. We can see the missing data distribution of O3 and pm2.5, and it is Quartile range (QR).

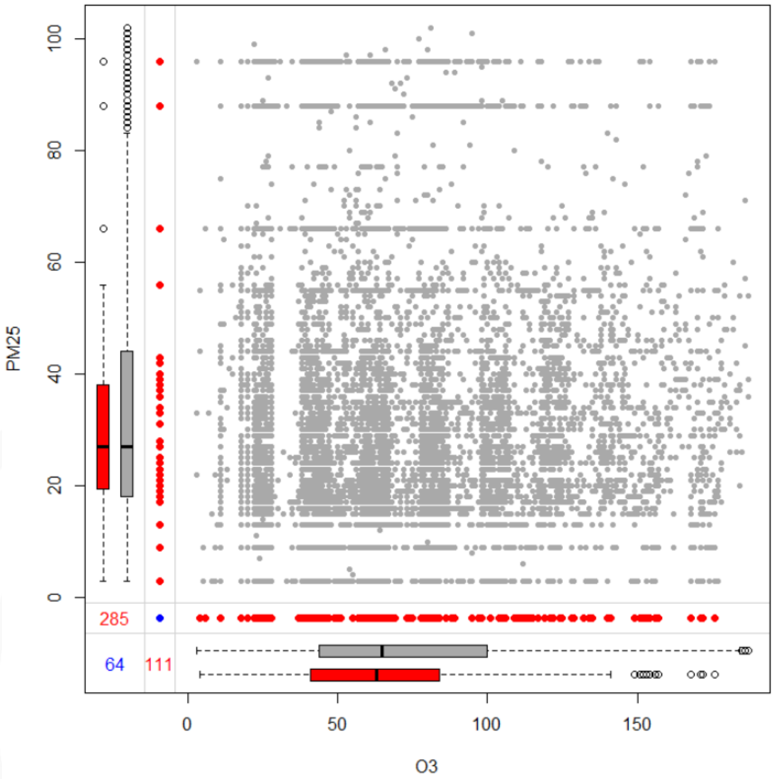


FIGURE 4.5: Missing Data Visualization of Distribution

Figure 4.6 is a visual distribution of the NA value to 0. It can be seen that the values are scattered in the red box.

Figure 4.7 is a visual distribution of the NA value to Mean. It can be seen that the values are scattered in the red box.

Figure 4.8 is a visual distribution of the NA value to 0; We can see the value of the distribution of the entire data.

4.3.2 Interpolation and Extrapolation

For the traditional methods of complementarity, simple experiments have found that there is no obvious help for accuracy, so choose other ways to make up the value.

The simplest of all the interpolation methods is the linear interpolation method. Any two adjacent table lists must be connected by a straight line so that the$x$)x

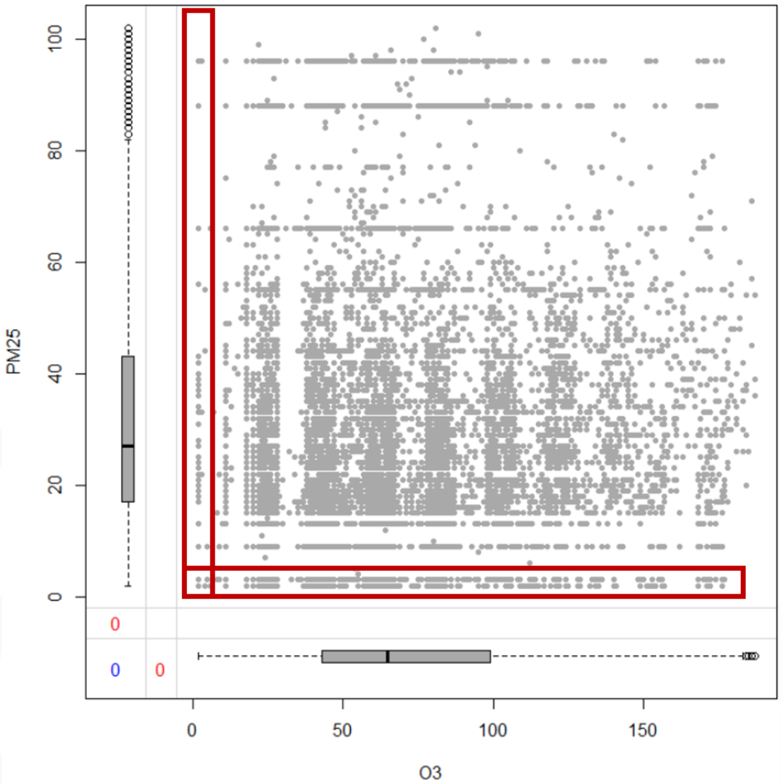


FIGURE 4.6: Using Zero to Fix Missing Data

value between them has a linear function $y(x)$. It can be used to use the property that the slope on the straight line must be a fixed value. The formula is (take (x_1, y_1), (x_2,y_2) for two adjacent table columns as an example):

$$(y - y_1)/(x - x_1) = (y_2 - y_1)/(x_2 - x_1)$$

After rearrange:

$$y = [(y_2 - y_1)/(x_2 - x_1)](x - x_1) + y_1$$

The right side of the equal sign is all x) and a constant. Therefore, we have a clear formula for $y(x)$ available.

In the early stages of the experiment was originally expected to use interpolation, but we use of interpolation will A problem is encountered. If the latest data is missing, it will not be able to be repaired, as shown in figure 4.9. The black points indicate the actual value, and the red one indicate the value after repair. Therefore, the Linear Entrapolation method was later used to compensate, and

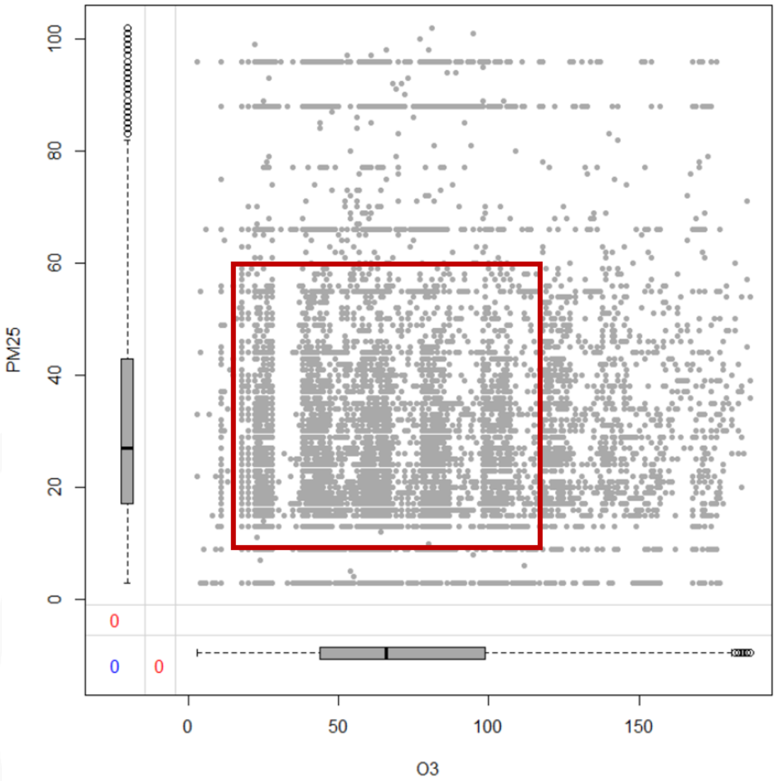


FIGURE 4.7: Using Mean to Fix Missing Data

the method has improved the accuracy of RNN training, as shown in figure 4.10.

4.3.3 Reduce Outliers

Through the first few charts, we discovered the problem of outliers. To further improve the accuracy, we will reduce the outliers from Quartile Range to improve the periodicity of the data and avoid outliers causing accuracy. Falling, and Auto-retraining needs performed multiple times.

Outliers, also known as escape values, mean that one or more values in the data are quite different from other values. The Chanwennt rule states that if the probability that a value deviates from the observed mean is less than or equal to $1/(2n)$, the data should be discarded (where n is the number of observations, and the probability can be estimated from the distribution of the data). When outliers occur, they must be handled with care. To combine professional knowledge and statistical methods, first check the raw data carefully to see if there is a reasonable

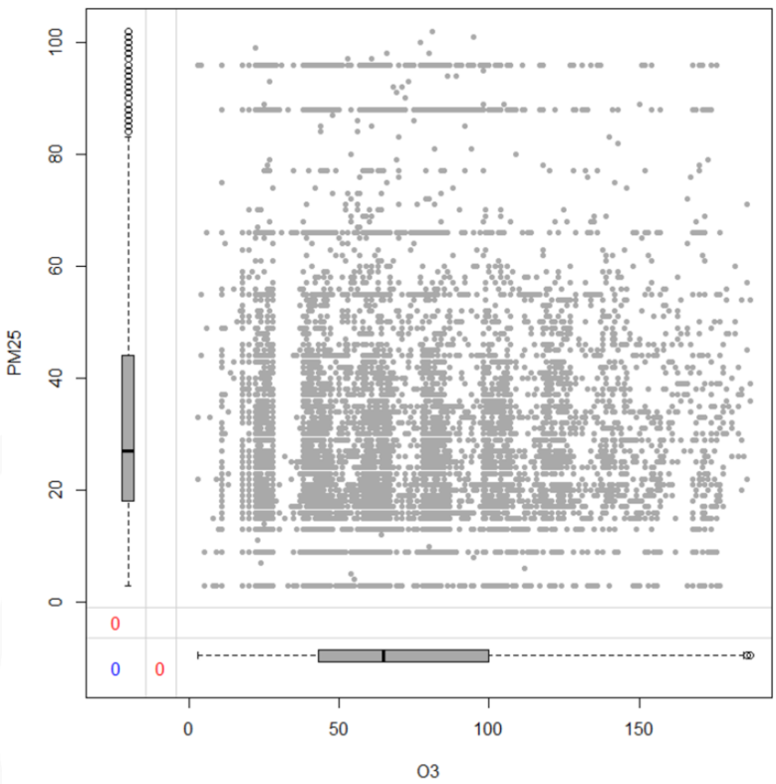


Figure 4.8: Using Random to Fix Missing Data

explanation for the profession. If this is the case, and it cannot be verified by finding the observation object, the observation value can only be deleted.

There are two main reasons for outliers:

1. The extreme manifestations of the overall are intrinsic variation. This is true and normal data. It is only extreme in this experiment that these outliers belong to the same population as the rest of the observations.

2. Due to the contingency of test conditions and experimental methods, or the results of observations, records, and calculation errors, it is an abnormal and erroneous data. These data do not belong to the same population as the rest of the observations.

Graph 4.12 is the result of reducing outliers through R's built-in quantile() function. By this method, we remove the extreme values and fill them with extrapolation.

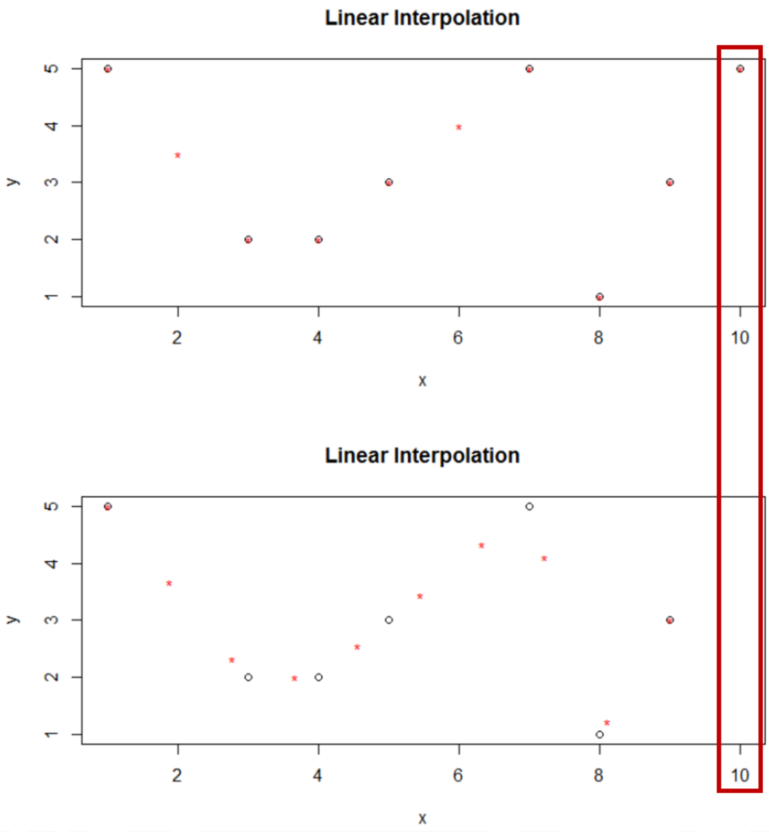


Figure 4.9: Linear Intrapolation

4.3.4 Results

This experiment uses 400 hours of data in Xitun District for complementation and 10 RNN training, as well as two random complements, as shown in Figure 4.13. It can be seen that extrapolation greatly improves the accuracy. Random results Time is good and bad, and the average method is the least accurate, as shown in Figure 4.14. Therefore, the RNN training of this experiment uses an extrapolation method for data repair.

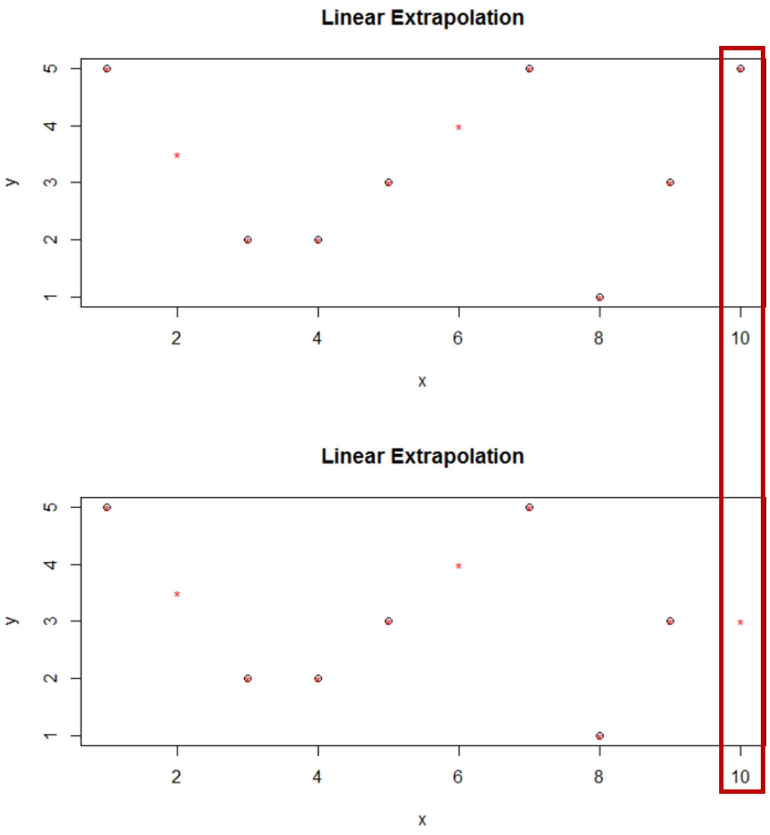


FIGURE 4.10: Linear Extrapolation

TABLE 4.3: Data of Training 10 Times using 4 Methods to Fix Missing Data

	NA	0	Mean	Random A	Random B	Linear Extrapolation
1	0.524951	0.441387	0.785425	0.228278	0.568871	0.2337555
2	0.608903	0.437471	0.814814	0.318034	0.492818	0.1993717
3	0.426354	0.37555	0.731502	0.33851	0.528137	0.2171834
4	0.484851	0.487092	0.818956	0.377991	0.52594	0.202295
5	0.578627	0.45561	0.901618	0.299305	0.527692	0.2501132
6	0.418446	0.585525	0.809997	0.274876	0.465154	0.1768674
7	0.52999	0.487943	0.679426	0.228497	0.592127	0.2706688
8	0.397076	0.460447	0.8232	0.29986	0.530278	0.3139126
9	0.550512	0.418908	0.726946	0.261572	0.528898	0.2462945
10	0.492725	0.425753	0.871231	0.190296	0.627671	0.2850107
Average	0.501244	0.457569	0.796312	0.281722	0.538759	0.23954728

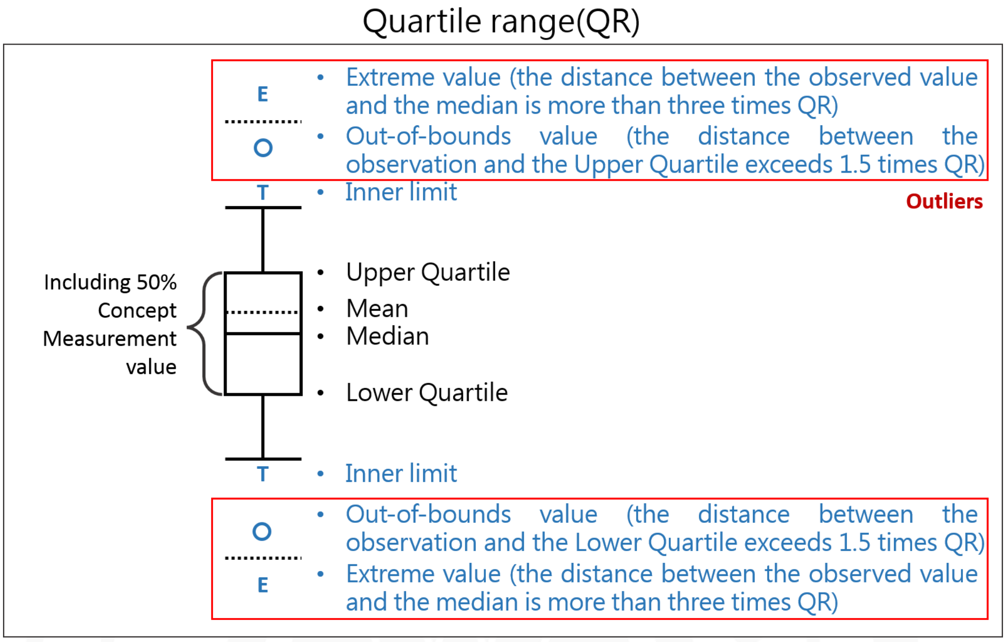

Quartile range(QR)
E
O
T
• Extreme value (the distance between the observed value and the median is more than three times QR)
• Out-of-bounds value (the distance between the observation and the Upper Quartile exceeds 1.5 times QR)
• Inner limit
Outliers
Including 50% Concept Measurement value
• Upper Quartile
• Mean
• Median
• Lower Quartile
T
• Inner limit
O
E
• Out-of-bounds value (the distance between the observation and the Lower Quartile exceeds 1.5 times QR)
• Extreme value (the distance between the observed value and the median is more than three times QR)

FIGURE 4.11: Quartile Range(QR)

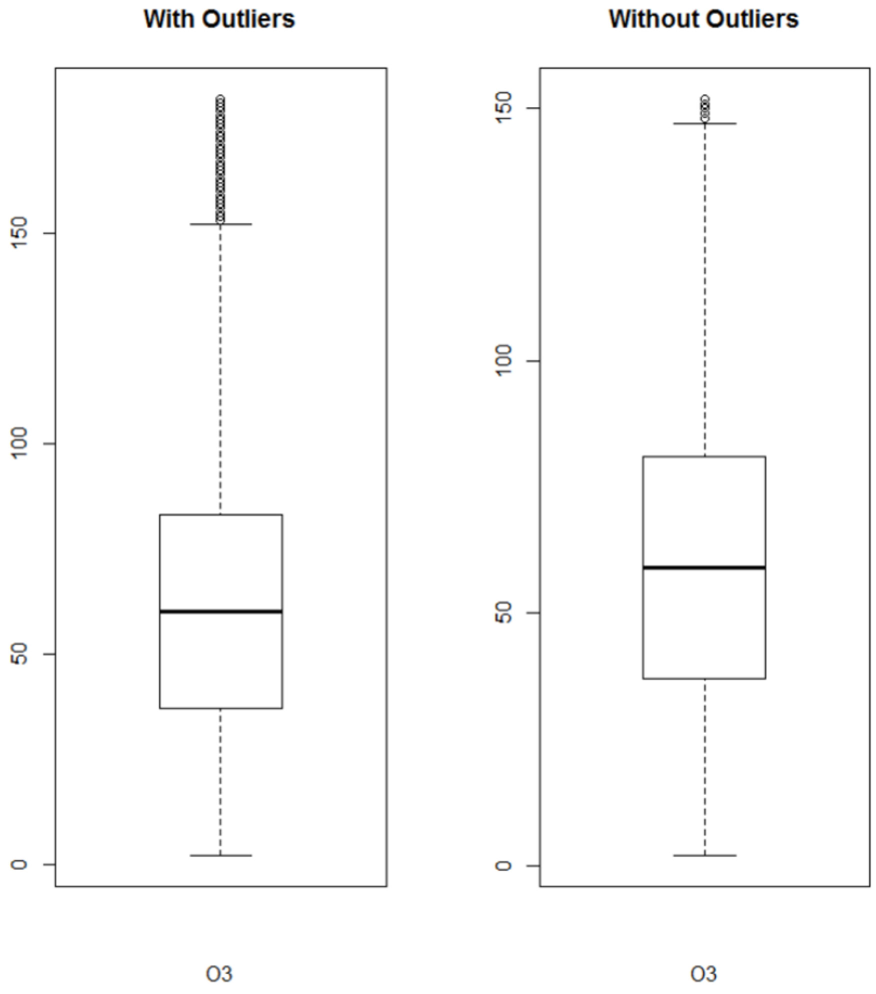


FIGURE 4.12: Reduce Outliers

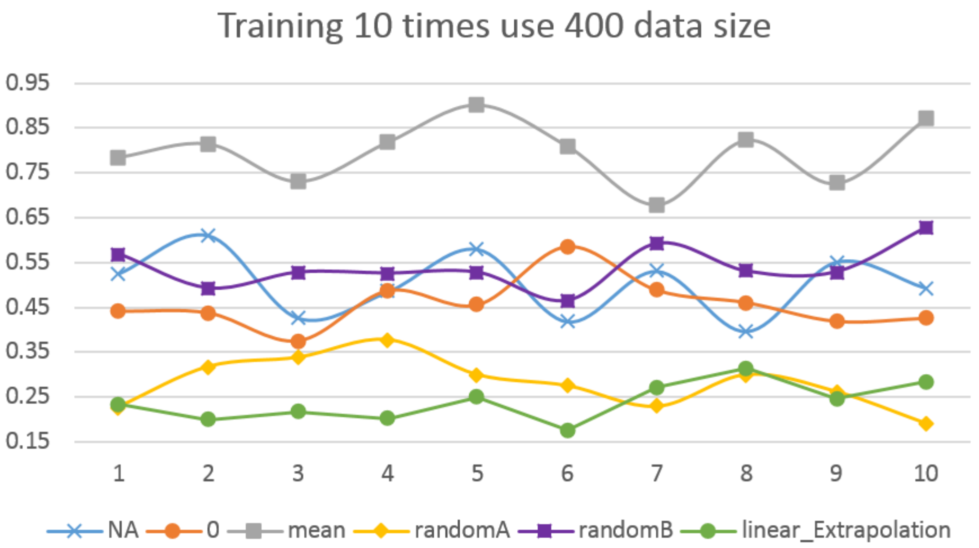


FIGURE 4.13: Training 10 Times using 4 Methods to Fix Missing Data

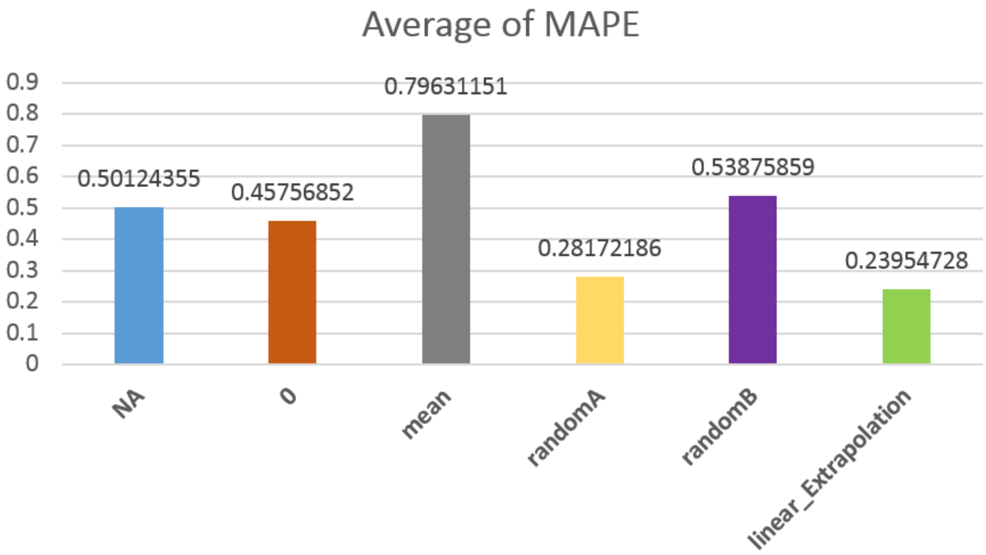


FIGURE 4.14: Average of Training 10 Times using 4 Methods

4.4 RNN Training for PM2.5 Prediction

4.4.1 Comparison of RNN and LSTM

Before deciding to use RNN, we compared the univariate prediction of RNN and LSTM. LSTM is an extension of RNN. The main difference is that multivariable training can be performed, but it is more complex than its algorithm, and it also requires more training time. In the experiment, 400 units of time data were divided into a training set of 360 units of time and a test set of 40 units of time, and data was repaired using extrapolation. Automatically detect the best Lag-Time and univariate O3 for 10 times of training in pm2.5. The result is shown in Figure 4.15 and Figure 4.16.

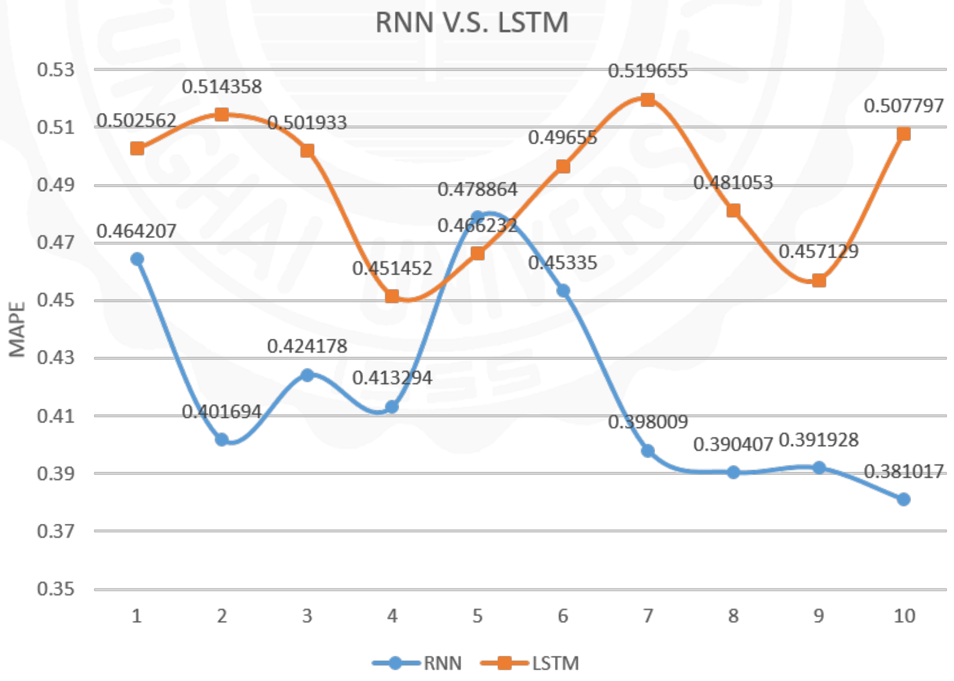


FIGURE 4.15: MAPE of RNN and LSTM Training 10 Times

In general terms, the LSTM effect is better than that of the RNN. The reason for this result is that the RNN parameters are optimized while the LSTM is not. Thus, the importance of parameter optimization to a set of modules can be seen.

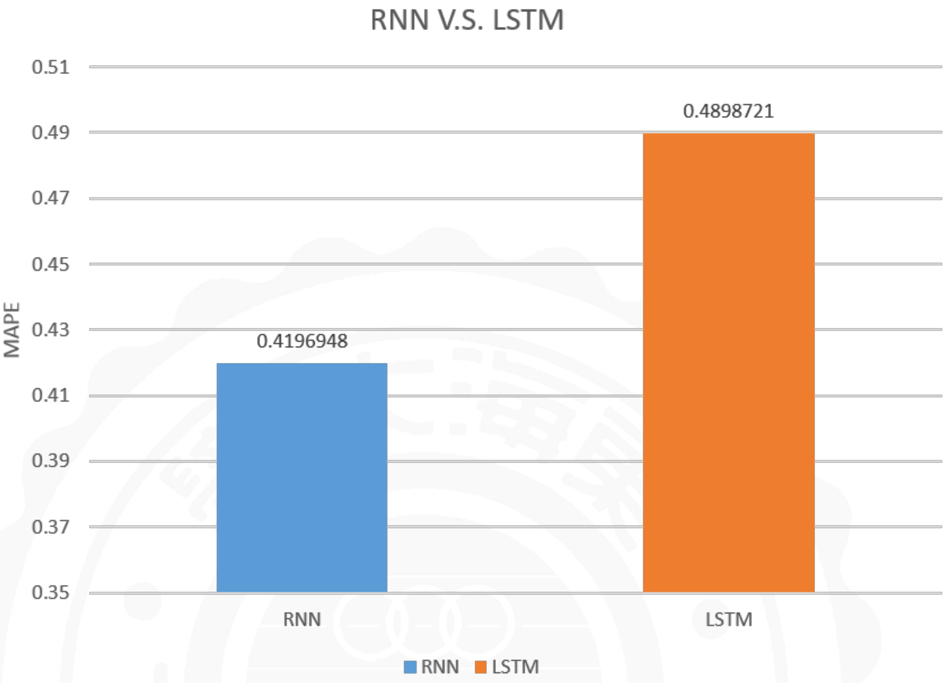


FIGURE 4.16: Average of RNN and LSTM Training 10 Times

4.4.2 RNN Parameters Optimization

To make the RNN parameters automatically detected, overall training time is much shorter than the data unit time (one hour). Because it is necessary to run RNN for the best model repeatedly, it will inevitably not consume too much time. The experiment is conducted through decentralized environment construction and optimization of the RNN program. The overall training time was much shorter than the data unit time. To make the RNN parameter automatically calibrated, we must first observe the relationship between the parameters. If one of the parameters and the remaining parameters of the parameter correlation is low, then this parameter can be considered into automatic calibration.

In the experiment of parameter and MAPE, it can be found that the Hidden layer and Learning rate are similar in the trend of different time series data for MAPE. Figure 4.17, figure 4.18, figure 4.19, figure 4.20, figure 4.21, and figure 4.22 show the parameter trend when the data size is 400 unit time. The Hidden layer is determined within the range of 10 to 20, and the learning rate is detemined within the range of 0.5 to 0.7. In general, the learning rate is between 0.5 and

0.8, so the experimental results are reasonable. Moreover, the effect of these two parameters on the MAPE value is not Lag-time, and the parameter Numepochs is large. So considering the training time, we directly set the Learning rate to 0.5 and the Hidden layer to 10.

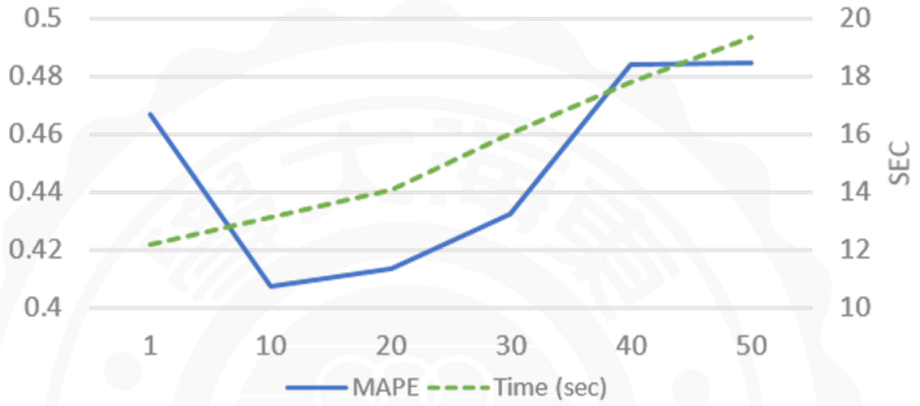


FIGURE 4.17: MAPE of Hidden Layer(1)

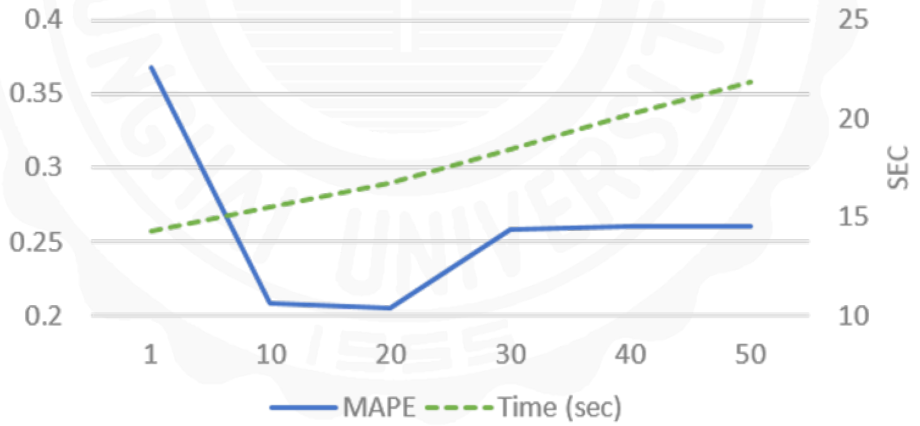


FIGURE 4.18: MAPE of Hidden Layer(2)

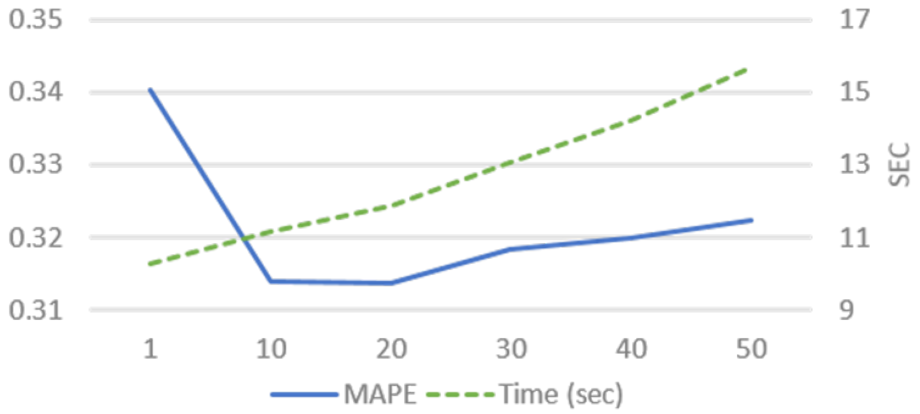


FIGURE 4.19: MAPE of Hidden Layer(3)

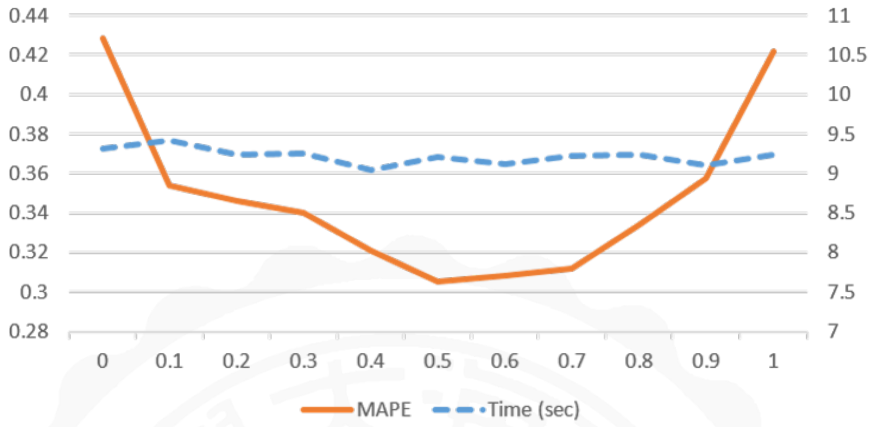


FIGURE 4.20: MAPE of Learning Rate(1)

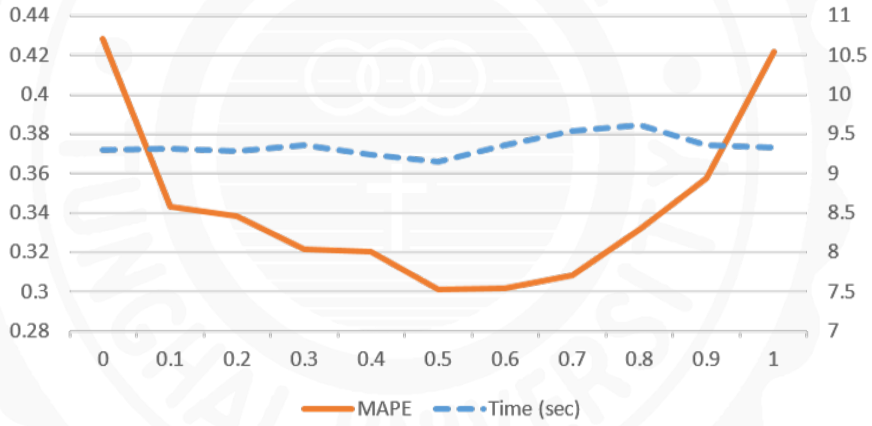


FIGURE 4.21: MAPE of Learning Rate(2)

In the experiment of parameter Numepochs, as Figure 4.23, Figure 4.24 and Figure 4.25, we found that different data ratios will cause MAPE to have different trends. We use the ‘optimizer accuracy’ in RNN to make it automatically end when we reach the ideal value we set it, to get the best model to provide more accurate predictions.

Because the optimal solution of parameter Numepochs is a floating value, we automatically stop it through the loss function in RNN to find the best accuracy value and stop automatically. The concept of the loss function is similar to setting a slope, and a point of the curve automatically stop when it reaches the set slope. The concept is shown in Figure 4.26.

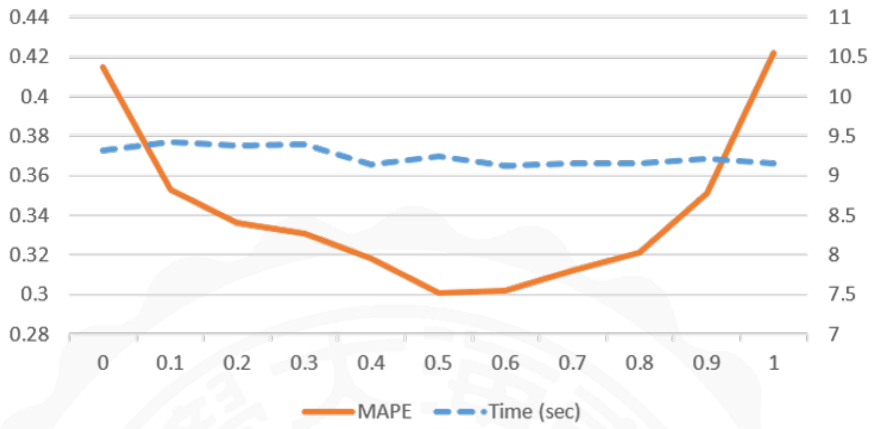


Figure 4.22: MAPE of Learning Rate(3)

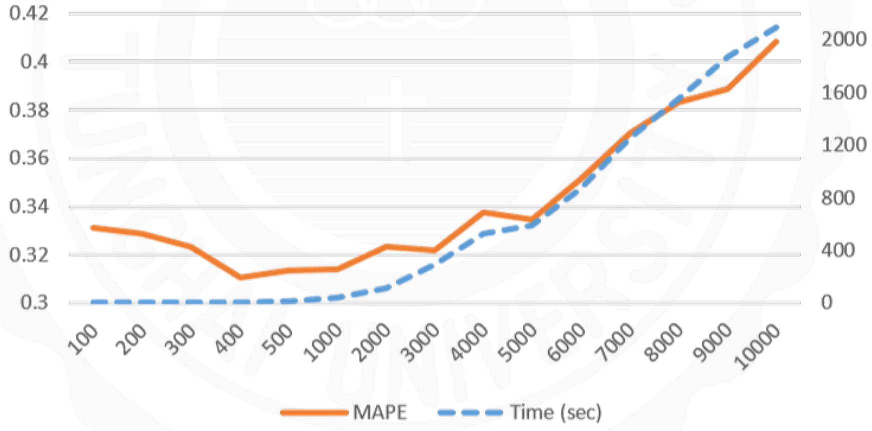


Figure 4.23: MAPE of Number of Epochs(1)

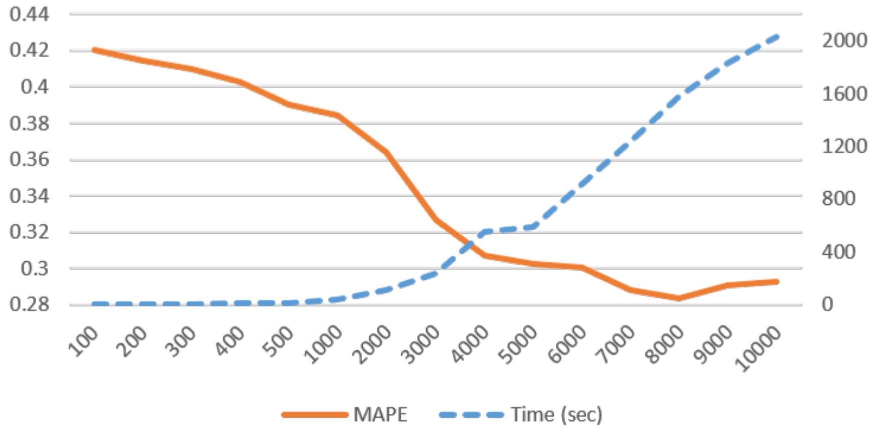


Figure 4.24: MAPE of Number of Epochs(2)

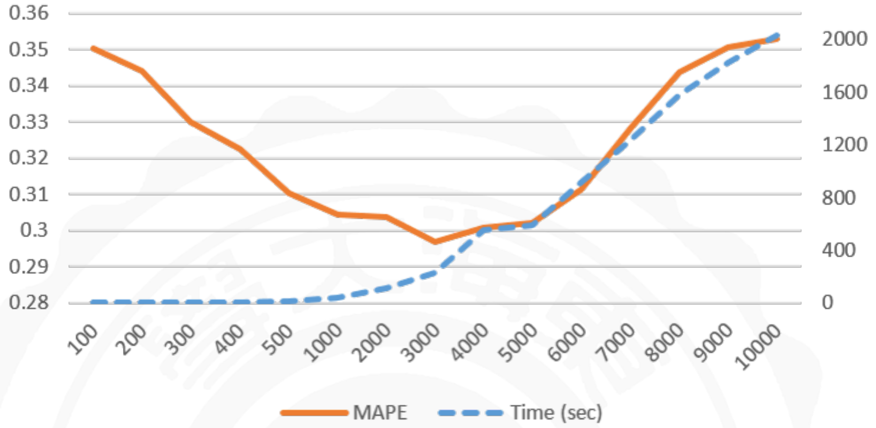


FIGURE 4.25: MAPE of Number of Epochs(3)

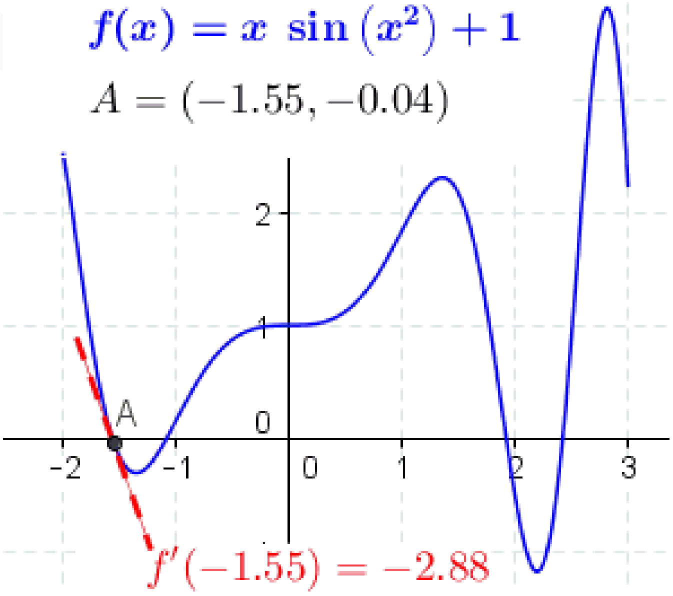
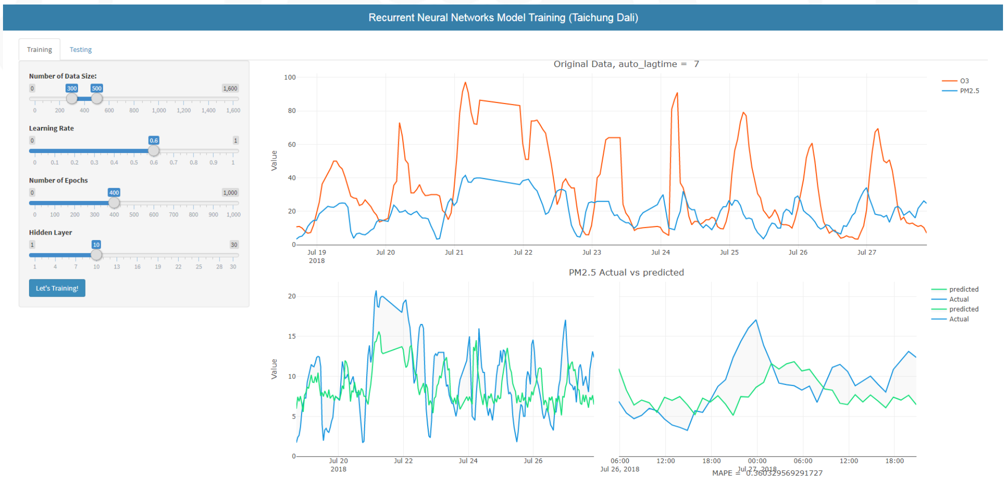


FIGURE 4.26: Use slope to find the optimal solution of the curve

4.4.3 RNN Model Visualization

In training the RNN model, we show the raw data, training data and test data on the shiny server interactive visualization chart, and can manually adjust the RNN parameters on its, as shown in Figure 4.27. This phase is to facilitate the preliminary analysis of the experiment and follow-up.

In the next step, on automatic adjustment of the parameters, we no longer need to select the parameter values manually, as shown in Figure 4.28. The top of the graph is the result of the automatic Lag-Time displacement of the original data O3 and PM2.5 [?]. To discard the data into the trained model and present the actual and predicted outcomes and their MAPE.


Recurrent Neural Networks Model Training (Taichung Dali)

Number of Data Size: 300 – 500

Learning Rate: 0.6

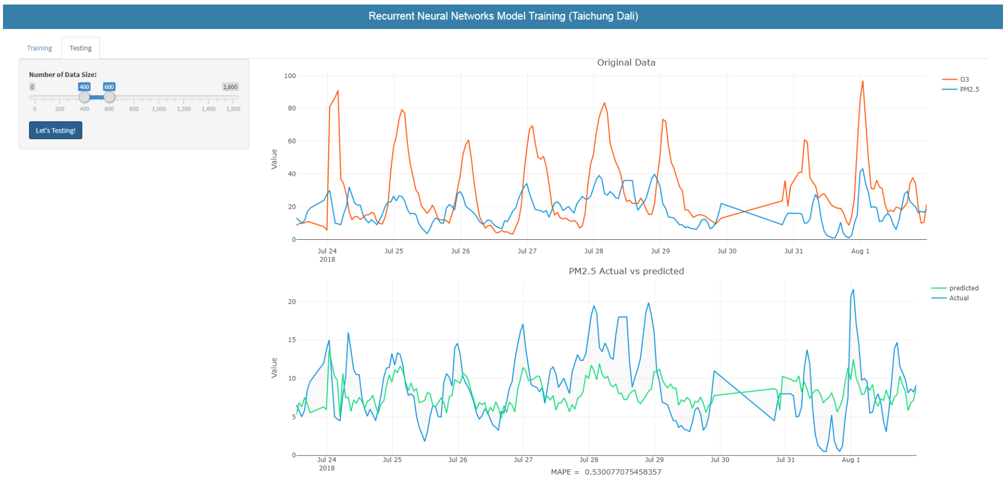
Number of Epochs: 400

Hidden Layer: 10

MAPE = 0.360329569291727

Figure 4.27: Shiny Website for Training

Figure 4.29 and Figure 4.30 show the results of training in different time zones in Xitun District, including the original values of PM2.5 and O3 before training, as well as training data and test data after training. Through RNN training, we can visualize through MAPE value. To quickly understand the influence of parameters on training accuracy. Then fine-tune the parameters and improve the module.



Recurrent Neural Networks Model Training (Taichung Dali)

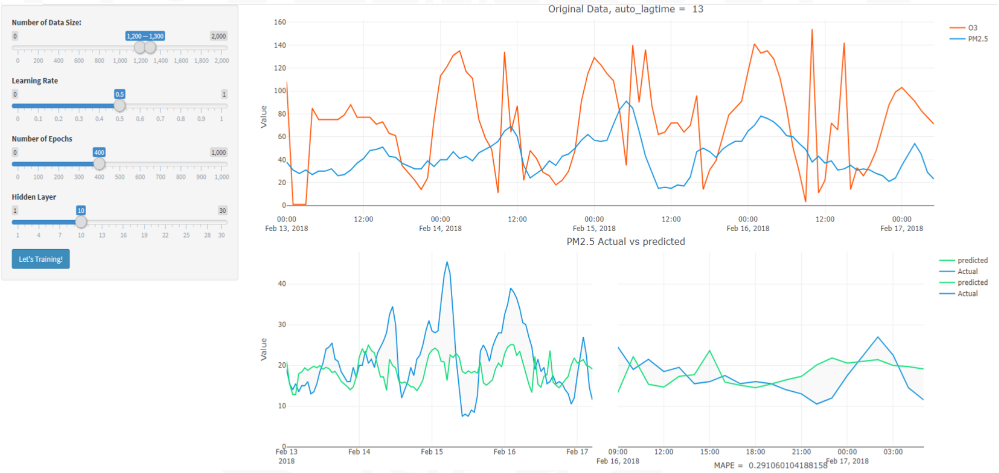
Training | Testing

Number of Data Size: 400 – 600

Let's Testing!

MAPE = 0.530077075458357

FIGURE 4.28: Shiny Website for Testing



Number of Data Size: 1,200 — 1,300

Learning Rate: 0.5

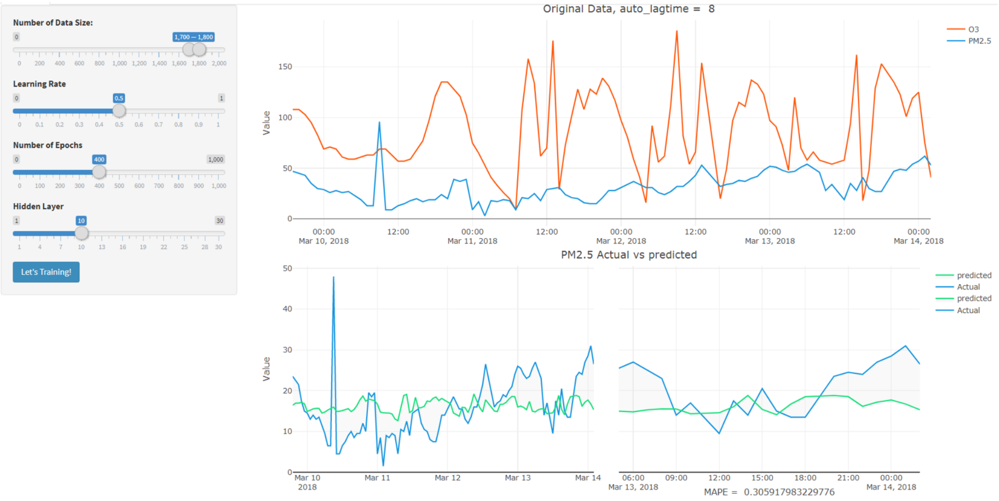
Number of Epochs: 400

Hidden Layer: 10

Let's Training!

MAPE = 0.291060104188158

FIGURE 4.29: Xitun Training Result (2018/02/13 to 2018/02/17)



Number of Data Size: 1,700 — 1,800

Learning Rate: 0.5

Number of Epochs: 400

Hidden Layer: 10

Let's Training!

MAPE = 0.305917983229776

FIGURE 4.30: Xitun Training Result (2018/03/10 to 2018/03/14)

4.5 Analysis of Forecast Results

4.5.1 Analysis Visualized Station Values

In the experiment, we mainly tested 16 stations, including five stations of Taiwan Environmental Protection Administration(EPA) and 11 stations of Taiwan Power Company(TPC).

After we can automatically train the model, we can set the schedule. When the MAPE is higher than the expected value, it will automatically retrain the model.

By patching Missing data, Lagtime is automatically calibrated, RNN module parameters are calibrated, and RNN automation is trained. Finally, the visualized graphs are presented and discussed , as shown in Figure 4.31.

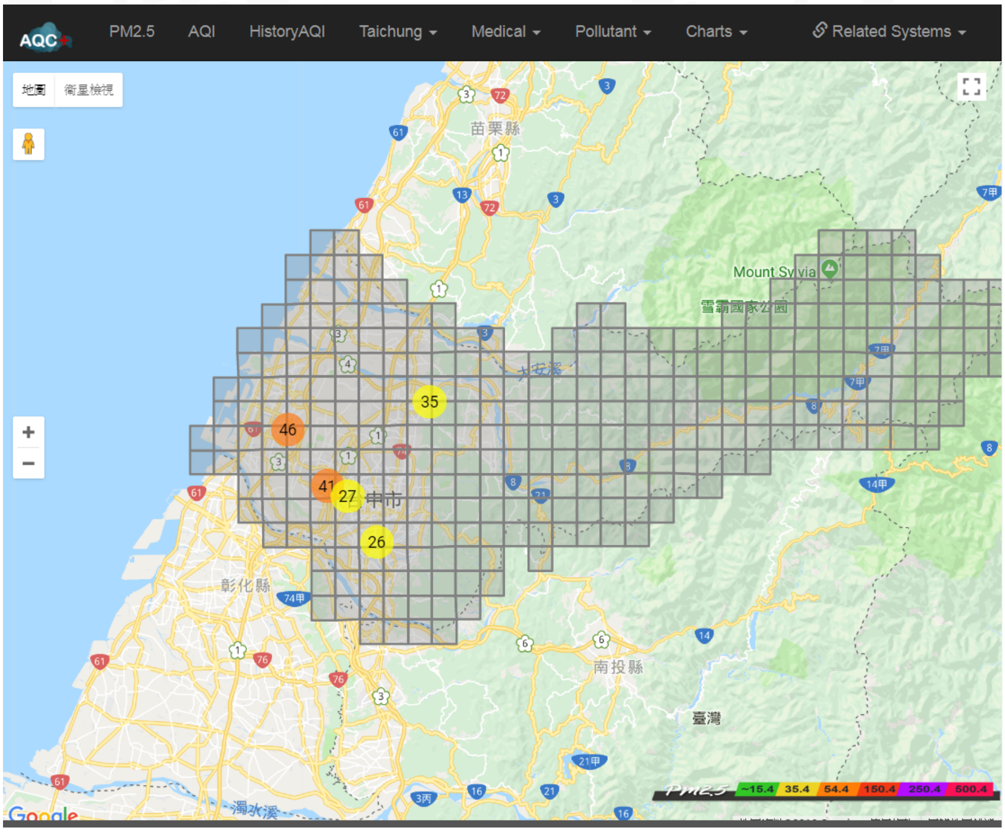


FIGURE 4.31: Website for Predict Visualization

It can be seen from figure 4.32 that the numerical fluctuation of the EPA station is more than that of the TPC. The reasons for this condition may be due to the position, type, and environment of the sensor.

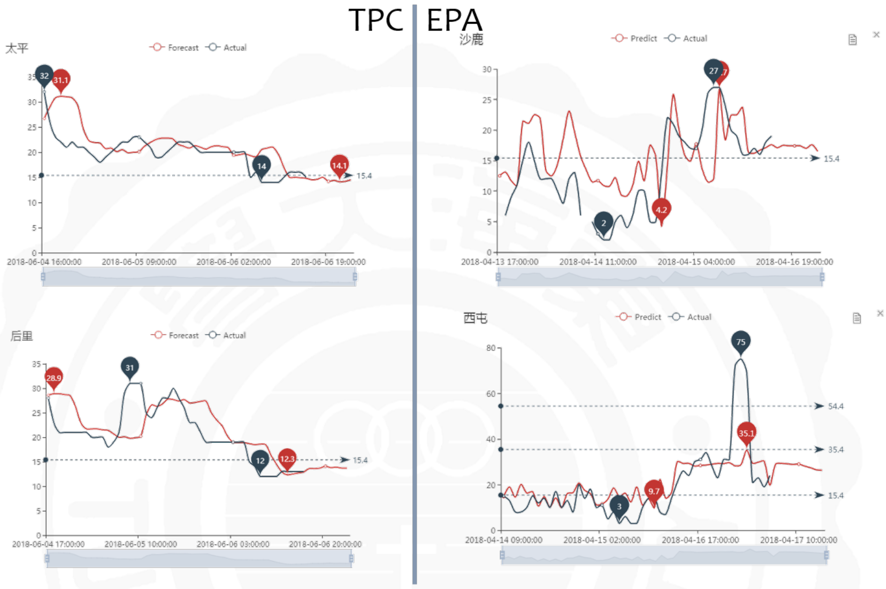


FIGURE 4.32: TPC stations and EPA stations

Then observe and analyze five EPA monitoring stations. The graph 4.33 is the PM2.5 value of the Dali monitoring station and its prediction result. The current training MAPE value falls between 0.3 and 0.5 on average, indicating that the predicted value has a reference value.

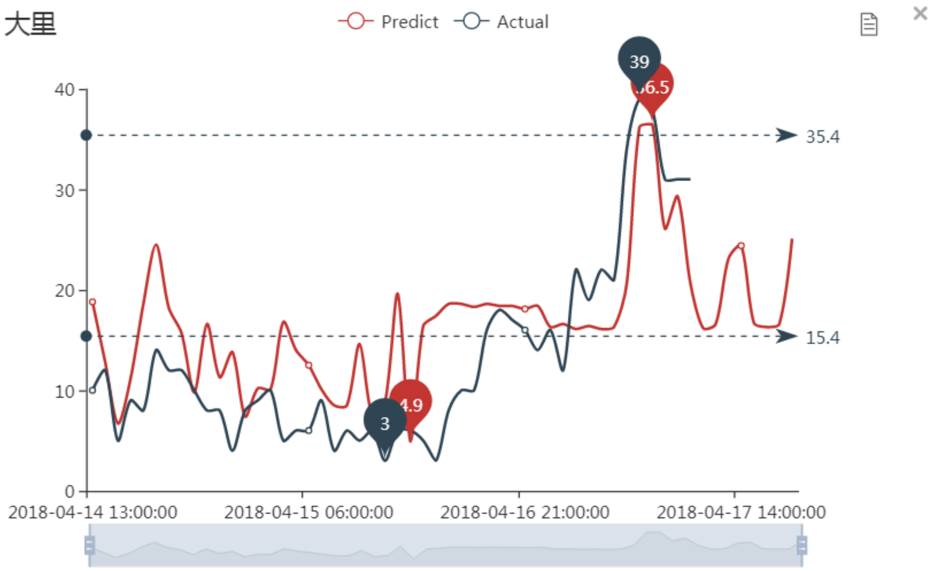


FIGURE 4.33: Prediction of Dali

It can be seen that both Fengyuan (Figure 4.34) and Shalu (Figure 4.35) have missing data, but because we do the extrapolation, we do not affect our prediction and accuracy.

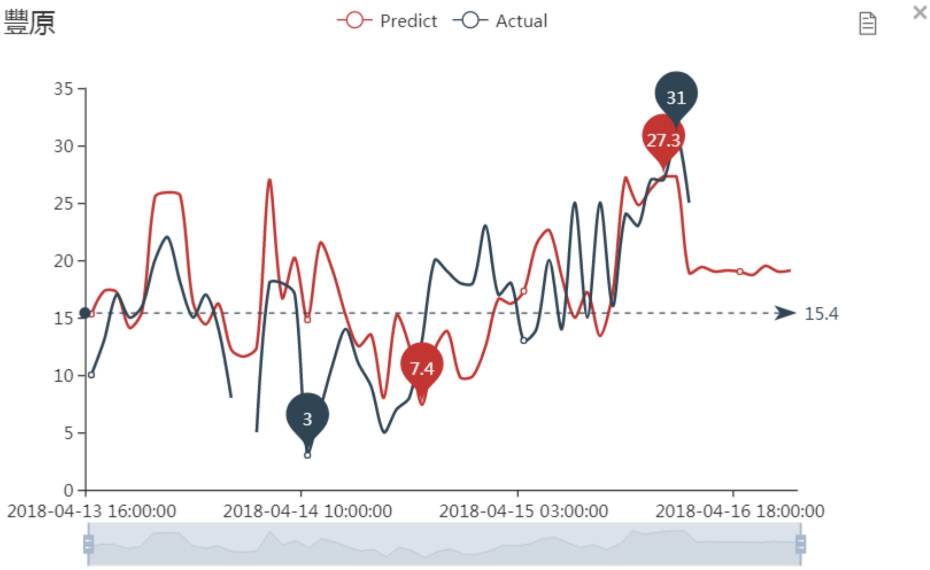


FIGURE 4.34: Prediction of Fengyuan

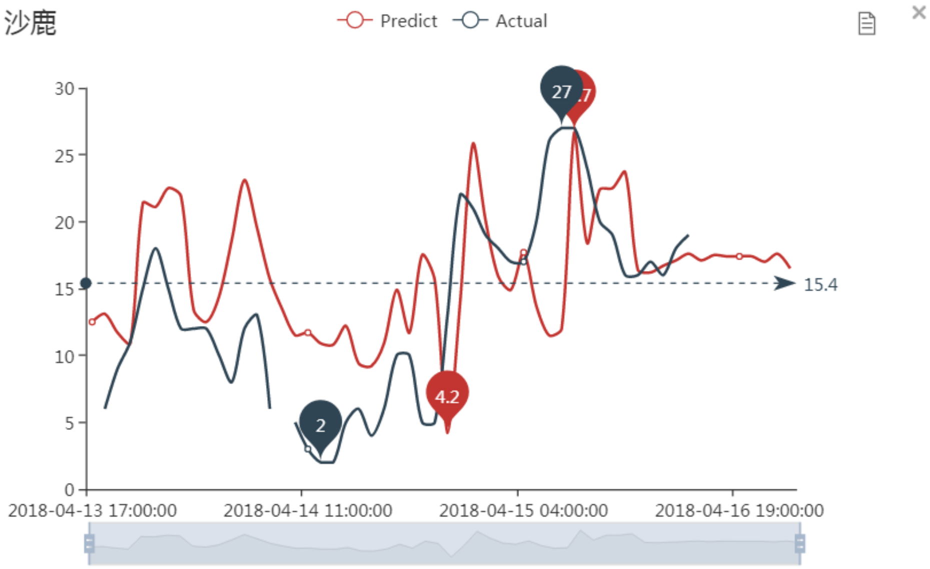


FIGURE 4.35: Prediction of Shalu

The Xitun monitoring station has a maximum value of 75 for one of the periods. PM2.5 unavoidably reduces the accuracy of the prediction. Therefore, extreme values must be removed and extrapolated using Linear Extrapolation. Avoids extreme values that necessitate automatic retraining.

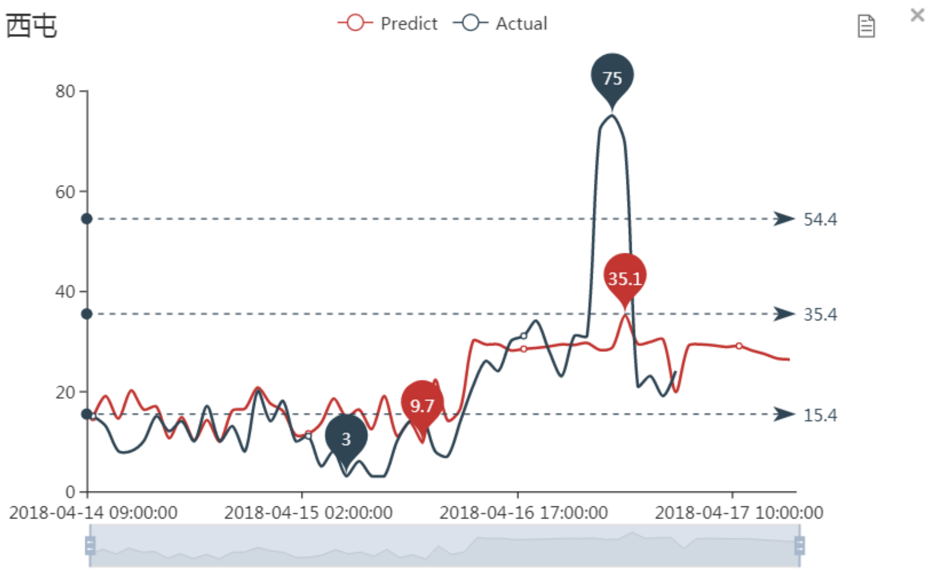


FIGURE 4.36: Prediction of Xitun

It can be seen that in addition to Loyalty's PM2.5 projections (Figure 4.37), Fengyuan, Dali, Xitun, and Shalu's predictions all have more than expected accuracy. It is presumed that the position and environmental factors of the Loyalty monitoring station may cause the correlation between PM2.5 and O3 values to be relatively insignificant. Therefore, it is impossible to apply the same module for prediction.

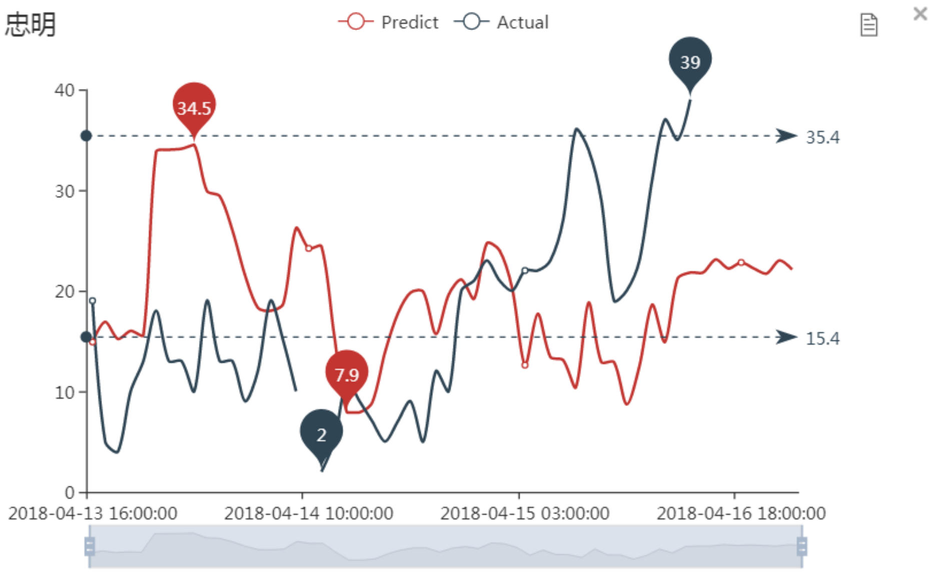


FIGURE 4.37: Prediction of Loyalty

Table 4.4: Retraining time statistics of EPA

2018/04/01 to 2018/04/30 | total 744hrs					
PublishTime	Fengyuan	Dali	Xitun	Shalu	Loyalty
2018/4/1 01:00	0	0	0	0	0
2018/4/1 02:00	0	0	0	0	0
2018/4/1 03:00	0	0	0	0	0
2018/4/1 04:00	0	0	0	0	0
2018/4/1 05:00	0	0	0	0	1
2018/4/1 06:00	1	1	0	0	0
2018/4/1 07:00	0	0	0	0	0
2018/4/1 08:00	0	0	0	0	0
2018/4/1 09:00	0	0	0	0	0
2018/4/1 10:00	0	0	0	1	1
...	...	...	...	...	...
Retraining times	41	35	32	37	61
Average	18.1463414	21.25714	23.25	20.10811	12.19672

4.5.2 Analysis RNN Model Retraining time

In addition to using visualization to analyze numerical fluctuations and predictive trends, we also recorded the frequency of retraining redesigns we designed. In addition to using visualization to analyze numerical fluctuations and predictive trends, we also recorded the frequency of retraining redesigns we designed.

At the time of retraining, we marked 1 and the rest 0. Finally, we added up the number of retrains. The total duration divided by the number of retraining hours. In Table 4.4, we can observe that the duration of retraining at the EPA station is about 20, and only the loyalty station falls at 12. After half a day, it needs to be retraining.

However, TPC stations will almost always be retraining for more than 24 hours, and most of the interesting retraining time fall in a similar period.

Table 4.5: Retraining time statistics of TPC

2018/04/01 to 2018/04/30 | total 744hrs					
PublishTime	Houli	Wenshan	Taiping	Qingshui	Wufeng
2018/4/1 01:00	0	0	0	0	0
2018/4/1 02:00	0	0	0	0	0
2018/4/1 03:00	0	0	0	0	0
2018/4/1 04:00	0	0	1	0	0
2018/4/1 05:00	0	0	0	0	0
2018/4/1 06:00	1	1	0	0	1
2018/4/1 07:00	0	0	0	0	0
2018/4/1 08:00	0	0	0	0	0
2018/4/1 09:00	0	0	0	0	0
2018/4/1 10:00	0	0	0	0	0
...	...	...	...	...	...
Retraining times	30	29	22	25	28
Average hrs	24.8	25.65517	33.8182	29.76	26.57142

In comparison between the two, we can see that TPC retraining time is almost always longer than that of the EPA. Apart from the fact that it also protects the fluctuations in numerical values, it may also be caused by differences in sensor position, height, and humidity.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

This work integrates the distributed system and the RNN module to predict PM2.5 in the air pollution data. Increase the correlation between O3 and PM2.5 through automatic detection of Lag-Time. Moreover, we observed the trend of the relationship between running time and MAPE through experiments from each parameter. Based on these experiments, we determined to set the parameters to a fixed value of 0.5 for the learning rate and 10 for the hidden layer. To make a more accurate prediction, we use the 'optimizer accuracy' in RNN which allow to detect them automatically. Therefore, the training module can adapt to the trend change of each segment and increase its prediction accuracy. Finally, by detecting the MAPE, the system recognizes the changes, then specify whether to retraining model or not. Therefore, this system gives beneficial effort as able to retraining the model automatically.

5.2 Future Works

In the future work, the main goal will be to increase the forecasting time and improve the accuracy of the training module. Compare the merits and demerits of different training modules for the air pollution data and further explore whether to use the training module to find the sensing. The abnormality of the instrument, such as its poor position and insufficient environment, helps improve the sensor's erection position.

References

[1] John C Brocklebank, David A Dickey, and Bong Choi. SAS for forecasting time series. SAS institute, 2018.

[2] LR Medsker and LC Jain. Recurrent neural networks. Design and Applications, 5, 2001.

[3] Jerome T Connor, R Douglas Martin, and Les E Atlas. Recurrent neural networks and robust time series prediction. IEEE transactions on neural networks, 5(2):240–254, 1994.

[4] Longjian Liu, Xuan Yang, Hui Liu, Mingquan Wang, Seth Welles, Shannon Márquez, Arthur Frank, and Charles N Haas. spatial–temporal analysis of air pollution, climate change, and total mortality in 120 cities of china, 2012–2013. Frontiers in public health, 4:143, 2016.

[5] Xiang Li, Ling Peng, Yuan Hu, Jing Shao, and Tianhe Chi. Deep learning architecture for air quality predictions. Environmental Science and Pollution Research, 23(22):22408–22417, 2016.

[6] Aowabin Rahman, Vivek Srikumar, and Amanda D. Smith. Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks. Applied Energy, 212:372 – 385, 2018.

[7] Vladimir Vapnik and Rauf Izmailov. Knowledge transfer in svm and neural networks. Annals of Mathematics and Artificial Intelligence, 81(1-2):3–19, 2017.

[8] R Daniel Meyer and Dianne Cook. Visualization of data. Current Opinion in Biotechnology, 11(1):89–96, 2000.

[9] Chris Beeley. Web application development with R using Shiny. Packt Publishing Ltd, 2013.

[10] David Kahle and Hadley Wickham. ggmap: Spatial visualization with ggplot2. R Journal, 5(1), 2013.

[11] Wen Zhang, Athena Eftychiou, Bogdan Vrusias, Nick Antonopoulos, Dongliang Zhang, Minhua Ma, Andreas Oikonomou, Nik Bessis, Peter Norrington, Yong Yue, et al. International journal of grid and high performance computing. International Standard Serial Number, 2014.

[12] Yun Bai, Zhenzhong Sun, Jun Deng, Lin Li, Jianyu Long, and Chuan Li. Manufacturing quality prediction using intelligent learning approaches: A comparative study. Sustainability, 10(1):85, 2017.

[13] Mohamed Akram Zaytar and CE El Amrani. Sequence to sequence weather forecasting with long short term memory recurrent neural networks. Int J Comput Appl, 143(11), 2016.

[14] Henrique do Nascimento Camelo, Paulo Sérgio Lucio, João Bosco Verçosa Leal Junior, Paulo Cesar Marques de Carvalho, and Daniel von Glehn dos Santos. Innovative hybrid models for forecasting time series applied in wind generation based on the combination of time series models with artificial neural networks. Energy, 151:347 – 357, 2018.

[15] Timothy M Dall, Paul D Gallo, Ritasree Chakrabarti, Terry West, April P Semilla, and Michael V Storm. An aging population and growing disease burden will require alarge and specialized health care workforce by 2025. Health affairs, 32(11):2013–2020, 2013.

[16] Dixian Zhu, Changjie Cai, Tianbao Yang, and Xun Zhou. A machine learning approach for air quality prediction: Model regularization and optimization. Big Data and Cognitive Computing, 2(1), 2018.

[17] Gaganjot Kaur Kang, Jerry Gao, Sen Chiao, Shengqiang Lu, and Gang Xie. Air quality prediction: Big data and machine learning approaches, 10 2017.

[18] Arnaud de Myttenaere, Boris Golden, Bénédicte Le Grand, and Fabrice Rossi. Mean absolute percentage error for regression models. Neurocomputing, 192:38 – 48, 2016. Advances in artificial neural networks, machine learning and computational intelligence.

[19] Jan Kleine Deters, Rasa Zalakeviciute, Mario Gonzalez, and Yves Rybarczyk. Modeling pm2. 5 urban pollution using machine learning and selected meteorological parameters. Journal of Electrical and Computer Engineering, 2017, 2017.

[20] Chao-Tung Yang, Yu-Wei Chan, Jung-Chun Liu, and Ben-Shen Lou. An implementation of cloud-based platform with r packages for spatiotemporal analysis of air pollution. The Journal of Supercomputing, pages 1–22, 2017.

[21] Emmanuel Paradis, Julien Claude, and Korbinian Strimmer. Ape: analyses of phylogenetics and evolution in r language. Bioinformatics, 20(2):289–290, 2004.

[22] Ross Ihaka and Robert Gentleman. R: a language for data analysis and graphics. Journal of computational and graphical statistics, 5(3):299–314, 1996.

[23] Mohd Dasari Anantha Reddy and Jabeed Rihaz. Importing data from mysql to hadoop using sqoop. Advances in Computational Sciences and Technology, 10(9):2835–2840, 2017.

[24] Chao-Hsien Lee and Yu-Lin Zheng. Automatic sql-to-nosql schema transformation over the mysql and hbase databases. In Consumer Electronics-Taiwan (ICCE-TW), 2015 IEEE International Conference on, pages 426–427. IEEE, 2015.

[25] Mei Liang, Cesar Trejo, Lavanya Muthu, Linh B Ngo, Andre Luckow, and Amy W Apon. Evaluating r-based big data analytic frameworks. In Cluster

Computing (CLUSTER), 2015 IEEE International Conference on, pages 508–509. IEEE, 2015.

[26] Roger S Bivand, Edzer J Pebesma, Virgilio Gómez-Rubio, and Edzer Jan Pebesma. Applied spatial data analysis with R, volume 747248717. Springer, 2008.

[27] Jin-Hee Ku. A study on prediction model of equipment failure through analysis of big data based on rhadoop. Wireless Personal Communications, 98(4): 3163–3176, 2018.

[28] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. Communications of the ACM, 51(1):107–113, 2008.

[29] Dhruba Borthakur et al. Hdfs architecture guide. Hadoop Apache Project, 53, 2008.

[30] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C Hsieh, Deborah A Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E Gruber. Bigtable: A distributed storage system for structured data. ACM Transactions on Computer Systems (TOCS), 26(2):4, 2008.

[31] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The Google file system, volume 37. ACM, 2003.

[32] Mr SS Aravinth, MS Shanmugapriyaa, MS Sowmya, and ME Arun. An efficient hadoop frameworks sqoop and ambari for big data processing. International Journal for Innovative Research in Science and Technology, 1(10): 252–255, 2015.

[33] Yangyang Xie, Bin Zhao, Lin Zhang, and Rong Luo. Spatiotemporal variations of pm2. 5 and pm10 concentrations between 31 chinese cities and their relationships with so2, no2, co and o3. Particuology, 20:141–149, 2015.

Appendix A

Hadoop-2.8.1 Installation

I. Modify hosts

```
# sudo vim /etc/hosts
```

II. Modify hostname

```
# sudo vim /etc/hostname
# sudo service hostname start
```

III. Install Java JDK

```
# sudo apt-get -y install openjdk-7-jdk
# sudo ln -s /usr/lib/jvm/java-7-openjdk-amd64 /usr/lib/jvm/jdk
```

IV. Add hadoop user

```
 # sudo addgroup hadoop
 # sudo adduser --ingroup hadoop hduser
 # sudo adduser hduser sudo
```

V. Creat SSH authentication login

```
# ssh-keygen -t rsa -f \~{}/.ssh/id\_{}rsa -P ""
# cp \~{}/.ssh/id\_{}rsa.pub ~/.ssh/authorized\_{}keys
# scp  – r ~/.ssh hduser:~/
```

VI. Download hadoop

```
# cd ~
# wget http://ftp.twaren.net/Unix/Web/apache/hadoop/common \\
   /hadoop-2.6.0/hadoop-2.6.0.tar.gz
# tar zxf hadoop-2.6.0.tar.gz
# mv hadoop-2.6.0.tar.gz hadoop
```

VII. Add the environment variable

```
# vim .bashrc

export JAVA_HOME=/usr/lib/jvm/jdk/
export HADOOP_INSTALL=/home/hduser/hadoop
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
```

VIII. Set hadoop config

```
# cd hadoop/etc/hadoop
# vim hadoop-env.sh

export JAVA_HOME=/usr/lib/jvm/jdk/

# vim core-site.xml

<property>
   <name>fs.default.name</name>
   <value>hdfs://hadoop-master:9000</value>
</property>

# vim yarn-site.xml

<property>
```

```
   <name>yarn.nodemanager.aux-services</name>
   <value>mapreduce_shuffle</value>
</property>
<property>
   <name>yarn.resourcemanager.hostname</name>
   <value>hduser</value>
</property>

# cp mapred-site.xml.template mapred-site.xml
# vim mapred-site.xml

<property>
   <name>mapreduce.framework.name</name>
   <value>yarn</value>
</property>

# mkdir -p ~/mydata/hdfs/namenode
# mkdir -p ~/mydata/hdfs/datanode
# vim hdfs-site.xml

<property>
   <name>dfs.replication</name>
   <value>2</value>
 </property>
 <property>
   <name>dfs.namenode.name.dir</name>
   <value>/home/hduser/mydata/hdfs/namenode</value>
 </property>
 <property>
   <name>dfs.datanode.data.dir</name>
   <value>/home/hduser/mydata/hdfs/datanode</value>
 </property>

# vim slaves
hadoop-master
node01
node02
```

IX. Copy hadoop to all nodes

```
# scp -r /home/hduser/hadoop node01:/home/hduser
# scp -r /home/hduser/hadoop node02:/home/hduser
```

X. Format HDFS

```
# hdfs namenode -format
```

XI. Start hadoop

```
# start-all.sh
```

XII. Use jps to see java running program

```
# jps
```

XIII. MapReduce JobTracker monitoring website

```
# hadoop-master:50030
```

Appendix B

HBase-0.95.2 Installation

I. Download HBase

```
# cd ~
# wget http://ftp.twaren.net/Unix/Web/apache/hbase\\
   /hbase-0.95.2/hbase-0.95.2-hadoop2-bin.tar.gz
```

II. Unzip hbase-0.95.2-hadoop2-bin.tar.gz

```
# tar zxf hbase-0.95.2-hadoop2-bin.tar.gz
```

III. Move the File of HBase

```
# mv hbase-0.95.2-hadoop2 hbase
```

IV. Set HBase config

```
# cd hbase
# vim conf/hbase-env.sh

export JAVA_HOME=/usr/lib/jvm/jdk
export HBASE_HOME=/home/hduser/hbase

# hadoop fs -mkdir /hbase
# vim conf/hbase-site.xml
```

```

<property>
        <name>hbase.rootdir</name>
        <value>hdfs://hadoop-master:9000/hbase</value>
</property>
<property>
        <name>hbase.cluster.distributed</name>
        <value>true</value>
</property>
<property>
        <name>hbase.zookeeper.quorum</name>
        <value>Test-master</value>
</property>

# vim conf/regionservers
hadoop-master
node01
node02
```

III. Copy jar to hbase/lib

```
# rm lib/hadoop-*
# cd /home/hduser/hadoop/share/hadoop
# cp *.jar /home/hduser/hbase/lib/
```

IV. Copy hbase to all nodes

```
# scp -r /home/hduser/hbase node01:/home/hduser
# scp -r /home/hduser/hbase node02:/home/hduser

# bin/start-hbase.sh
```

V. HBase monitoring website

```
# hduser:60010
```

Appendix C

Sqoop-1.4.6 Installation

I. Download Apache Sqoop

```
# sudo wget http://apache.stu.edu.tw/sqoop/1.4.6/sqoop-1.4.6.tar.gz
```

II. Unzip sqoop-1.4.6.tar.gz

```
# tar -zxvf sqoop-1.4.6.tar.gz
# mv sqoop-1.4.6/ sqoop
```

III. Setting .bashrc

```
# sudo vim ~/.bashrc
export SQOOP_HOME=/home/hduser/sqoop
export SQOOP_CONF_DIR="$SQOOP_HOME/conf"
export SQOOP_CLASSPATH="$SQOOP_CONF_DIR"
export PATH=$PATH:$SQOOP_HOME/bin
# source ~/.bashrc
```

IV. Create empty accumulo directory

```
# sudo mkdir accumulo
# sudo chown hadoop:hduser accumulo
```

V. Create and edit sqoop-env.sh

```
# sudo cp sqoop/conf/sqoop-env-template.sh sqoop/conf/sqoop-env.sh
# sudo vim sqoop/conf/sqoop-env.sh
export HADOOP_COMMON_HOME=/home/hduser/hadoop
export HADOOP_MAPRED_HOME=/home/hduser/hadoop
export HCAT_HOME=/home/hduser/hive/hcatalog
export HBASE_HOME=/home/hduser/hbase
export HIVE_HOME=/home/hduser/hive
export ACCUMULO_HOME=/home/hduser/accumulo
```

VI. Download MySql connector

```
# wget http://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-5.1.36.zip
# sudo mv mysql-connector-java-5.1.36-bin.jar sqoop/lib
```

Appendix D

Prediction Script Code

D.1 MainFunction.R Code

```
library("RMySQL"    , lib.loc="/usr/local/lib/R/site-library")
library("rnn"       , lib.loc="/usr/local/lib/R/site-library")
library("rlist"     , lib.loc="/usr/local/lib/R/site-library")

### Require function
source("/srv/shiny-server/sample-apps/RNN/prediction/func/mainf.R")
source("/srv/shiny-server/sample-apps/RNN/prediction/func/rnn_model.R")
source("/srv/shiny-server/sample-apps/RNN/prediction/func/auto_lagtime.R")
source("/srv/shiny-server/sample-apps/RNN/prediction/func/linear_extrapolation.R")
source("/srv/shiny-server/sample-apps/RNN/prediction/func/retraining.R")

### Connect to MySQL db
conn <- dbConnect(MySQL(),dbname = "Environment", username="root",password="hpc123",
    host="140.128.101.210")
dbSendQuery(conn,"SET NAMES utf8")
# Get the air quality data
senser_taichung_db <- dbGetQuery(conn, "select * from airquality WHERE County ='臺中市
    '")
# Catch the 16 sitenames from air quality db
senser_sitename_df <- as.data.frame(table(senser_taichung_db$SiteName))
senser_sitename_df <- data.frame(sitename=senser_sitename_df[(which(nchar(as.vector(
    senser_sitename_df$Var1)) < 3)),][,-2])
#x=1
### Get the data of every sitenames (500 hours)
for(x in c(1:length(senser_sitename_df[,1]))){
  # Use sitename to get the data from mysql db
  sitename<-as.character(senser_sitename_df[x,])
  sitename_str <- paste0("select Sitename,O3,PM25,PublishTime from airquality WHERE
    Sitename ='",sitename,"' ORDER BY ID DESC limit 500")
  senser_db <- dbGetQuery(conn,sitename_str)
  # Call linear_Extrapolation & main_data_clean functions
  senser_clear_db<-main_data_clean(linear_Extrapolation(senser_db[,-1]))
  # Combine every db to one list
  if (x ==1){
    sensor_list<-list(list(Sitename=senser_sitename_df[x,1],PM25_t=
    senser_clear_db$PM25_t,O3_t=senser_clear_db$O3_t,Time=senser_clear_db$Time,O3_p=
    senser_clear_db$O3_p,Time_O3_p=senser_clear_db$Time_O3_p,lagtime=
    senser_clear_db$lagtime))
  }else{
    sensor_list[[length(sensor_list)+1]] <- list(list(Sitename=senser_sitename_df[x
    ,1],PM25_t=senser_clear_db$PM25_t,O3_t=senser_clear_db$O3_t,Time=
    senser_clear_db$Time,O3_p=senser_clear_db$O3_p,Time_O3_p=senser_clear_db$Time_O3_p
    ,lagtime=senser_clear_db$lagtime))
  }
}
# Change format for list
sensor_list_df <- lapply(sensor_list, data.frame, stringsAsFactors = FALSE)

### Do training & prediction
for(x in c(1:length(senser_sitename_df[,1]))){

  sensor_not_list_df<-as.data.frame(sensor_list_df[x])
  ttable<-data.frame(Sitename=sensor_not_list_df$Sitename,PM25_t=
    sensor_not_list_df$PM25_t,O3_t=sensor_not_list_df$O3_t,Time=
    sensor_not_list_df$Time,O3_p=sensor_not_list_df$O3_p,Time_O3_p=
    sensor_not_list_df$Time_O3_p,lagtime=sensor_not_list_df$lagtime)
  senser_predict_df<-retraining(ttable)

  PM25_pred_df <- data.frame(PM25_pred=senser_predict_df$PM25_p,Time=ttable$Time_O3_p,
    sitename=ttable[x,1])
  #PM25_pred_df$PM25_pred<-abs(PM25_pred_df$PM25_pred-(abs(median(tail(ttable,8)
    $PM25_t)-tail(tail(PM25_pred_df,ttable[1,4]+1),1)$PM25_pred))+runif(1, -1.5, 1.5))
  PM25_pred_df$PM25_pred<-PM25_pred_df$PM25_pred+abs(median(tail(ttable,8)$PM25_t)-
    tail(tail(PM25_pred_df,ttable[1,6]+1),1)$PM25_pred)

  # Get the last 8 hours data
  l_fen<-1:8+length(PM25_pred_df[,1])-head(ttable$lagtime,1)
  PM25_pred_df<-PM25_pred_df[l_fen,]

  # Combine every data to one list
  if (x ==1){
    sensor_list<-list(list(PM25_pred=PM25_pred_df$PM25_pred ,Time=PM25_pred_df$Time ,
    sitename=PM25_pred_df$sitename))
  }else{
    sensor_list[[length(sensor_list)+1]] <-list(list(PM25_pred=PM25_pred_df$PM25_pred
    ,Time=PM25_pred_df$Time ,sitename=PM25_pred_df$sitename))
  }
}
# Change format for list
sensor_list_df <- lapply(sensor_list, data.frame, stringsAsFactors = FALSE)

for(x in c(1:length(senser_sitename_df[,1]))){

  if (x ==1){
    PM25_pred_df <- rbind(as.data.frame(sensor_list_df[x]))
  }else{
    PM25_pred_df <- rbind(as.data.frame(sensor_list_df[x]),PM25_pred_df)
  }
}

### Save back to mysql db
conn_rnn <- conn#dbConnect(MySQL(),dbname = "rnn", username="root",password="hpc123",
    host="140.128.101.210")
dbSendQuery(conn_rnn,"SET NAMES utf8")
# History Data Save to MySQL
PM25_Taichung_4hrs_pred_history_db <- dbGetQuery(conn_rnn, "SELECT PM25_pred,Time,
    sitename FROM PM25_Taichung_4hrs_pred_history_16")
PM25_Taichung_4hrs_pred_history_df <- data.frame(PM25_Taichung_4hrs_pred_history_db)
PM25_pred_hisrtory_df <- rbind(PM25_pred_df
                              ,PM25_Taichung_4hrs_pred_history_df
)
# Remove Duplicate data form db
ts<-c(2,3)
PM25_pred_hisrtory_df<-PM25_pred_hisrtory_df[!duplicated(PM25_pred_hisrtory_df[,ts]),
    ]
dbWriteTable(conn_rnn ,"PM25_Taichung_4hrs_pred_history_16",PM25_pred_hisrtory_df, row
    .names=FALSE,overwrite = TRUE)
```

D.2 AutoLagtime.R Code

```
# Auto_lagtime function ----------------------------------------
auto_lagtime<-function(senser_df) {
  # auto lagtime
  lagtime_24hr<-c(8:48)

  #lagtime_for_pm25<-5:28
  lagtime_for_pm25 <- 48:(length(senser_df[,1])-47)
  lagtime_PM25_df  <- senser_df[lagtime_for_pm25,]
  lagtime_PM25_df  <- lagtime_PM25_df[order(lagtime_PM25_df[,1],decreasing = TRUE),]
  lagtime_max_PM25 <- as.numeric(matrix(rownames(lagtime_PM25_df[]), nrow = length(
    lagtime_PM25_df[,1]))[1])

  lagtime_for_O3   <- lagtime_max_PM25-lagtime_24hr
  lagtime_O3_df    <- senser_df[lagtime_for_O3,]
  lagtime_O3_df    <- lagtime_O3_df[order(lagtime_O3_df[,2],decreasing = TRUE),]
  lagtime_max_O3   <- as.numeric(matrix(rownames(lagtime_O3_df[]), nrow = length(
    lagtime_O3_df[,1]))[1])

  lagtime <- lagtime_max_PM25-lagtime_max_O3
  #class(lagtime)
  return(lagtime)
}
```

D.3 LinearExtrapolation.R Code

```
library("Hmisc" , lib.loc="/usr/local/lib/R/site-library")
linear_Extrapolation <- function(senser_db){

  senser_db$PM25[which(senser_db$PM25  %in% c("ND","",NA,0))] <- NA
  senser_db$O3[which(senser_db$O3  %in% c("ND","",NA,0))] <- NA

  senser_NA_db<-senser_db

  senser_NA_db_length <- c(1:length(senser_NA_db[,1]))
  O3 <- as.numeric(senser_NA_db$O3)
  PM25 <-as.numeric(senser_NA_db$PM25)
  senser_linear_Extrapolation_db<-senser_NA_db
  senser_linear_Extrapolation_db$O3   <- as.data.frame(approxExtrap(
    senser_NA_db_length , O3,xout=senser_NA_db_length))$y
  senser_linear_Extrapolation_db$PM25 <- as.data.frame(approxExtrap(
    senser_NA_db_length , PM25,xout=senser_NA_db_length))$y

  return(senser_linear_Extrapolation_db)
}
```

D.4 RemoveOutliers.R Code

```
remove_outliers <- function(x, na.rm = TRUE, ...) {
  qnt <- quantile(x, probs=c(.25, .75), na.rm = na.rm, ...)
  H <- 1.5 * IQR(x, na.rm = na.rm)
  y <- x
  y[x < (qnt[1] - H)] <- NA
  y[x > (qnt[2] + H)] <- NA
  y
}
```

D.5 Retraining.R Code

```
library("rnn"      , lib.loc="/usr/local/lib/R/site-library")
library("rlist"    , lib.loc="/usr/local/lib/R/site-library")
retraining<-function(senser_train_df
                     ,learning_rate = 0.5
                     ,hidden_dim = 12
                     ,num_epochs = 300)
{
  rnn_t_df_move <- data.frame(PM25=senser_train_df[,1],O3=senser_train_df[,2],Time=
    senser_train_df[,4])
  O3_o   <- matrix((rnn_t_df_move$O3)/2, nrow = 5)
  PM25_o <- matrix((rnn_t_df_move$PM25)/2, nrow = 5)

  Omax <- max(PM25_o)
  Omin <- min(PM25_o)

  # Standardize in the interval 0 - 1
  O3_t   <- (O3_o - min(O3_o)) / (max(O3_o) - min(O3_o))
  PM25_t <- (PM25_o - min(PM25_o)) / (max(PM25_o) - min(PM25_o))

  # Transpose
  O3_t   <- t(O3_t)
  PM25_t <- t(PM25_t)

  # Training-testing sets
  tnum  <- length(rnn_t_df_move[,1])/100
  train <- 1:tnum
  test  <- (tnum+1):(length(rnn_t_df_move[,1])/5)
  time_test_range <- (length(rnn_t_df_move[,1])/5*4+1):length(rnn_t_df_move[,1])

  # Train model. Keep out the last two sequences.
  model<-model(O3_t[train,],PM25_t[train,],learning_rate,hidden_dim,num_epochs)

  # Predicted values
  PM25_p  <- predict(model, O3_t)
  PM25_pb <- PM25_p*(Omax-Omin)+Omin

  # MAPE 1hr to 8hrs
  actual <- data.frame(as.vector(PM25_o[,test]))
  pred   <- data.frame(as.vector(t(PM25_pb[test,])))
  mape   <- mean(abs((actual[,1]-pred[,1])/actual[,1]))
  mape

  PM25_pred_result<-data.frame(PM25_p=pred$as.vector.t.PM25_pb.test.....)

  return(PM25_pred_result)
}
```

D.6 DataPrediction.R Code

```
# Main_Data_Clean-----------------------------------------------------------------
source("/srv/shiny-server/sample-apps/RNN/prediction/func/wash_data.R")
source("/srv/shiny-server/sample-apps/RNN/prediction/func/auto_lagtime.R")
main_data_clean<-function(senser_db,num=400)
{
  senser_df<-wash_data(senser_db)

  # Call auto_lagtime function
  auto_lagtime<-auto_lagtime(senser_df)

  # Get the last Time (100hrs)
  hrs_100 <- ((length(senser_df[,1])-num+1):length(senser_df[,1]))

  # For Pred
  senser_Fengyuan_O3_pred   <- data.frame( O3  =senser_df[hrs_100,2],Time=senser_df[
    hrs_100,3]+(auto_lagtime*60*60))
  # For Train
  senser_Fengyuan_O3_ture   <- data.frame( O3  =senser_df[hrs_100-auto_lagtime,2],Time
    =senser_df[hrs_100-auto_lagtime,3]+auto_lagtime*60*60 )
  senser_Fengyuan_PM25_ture <- data.frame( PM25=senser_df[hrs_100,1],Time=senser_df[
    hrs_100,3] )

  senser_PM25_O3_df<-data.frame(PM25_t=senser_Fengyuan_PM25_ture$PM25
                                ,O3_t=senser_Fengyuan_O3_ture$O3
                                ,Time=senser_Fengyuan_O3_ture$Time
                                ,O3_p=senser_Fengyuan_O3_pred$O3
                                ,Time_O3_p=senser_Fengyuan_O3_pred$Time
                                ,lagtime=auto_lagtime)
  return(senser_PM25_O3_df)
}
```

D.7 WashData.R Code

```
wash_data<-function(senser_db){
  #senser_db<-senser_real
  senser_db <- senser_db[order(senser_db$PublishTime),]
  senser_df <- data.frame(PM25=senser_db$PM25,O3=senser_db$O3,Time=
    senser_db$PublishTime)

  # Data format
  senser_df$Time <- factor(senser_df$Time)
  senser_df$Time <- as.POSIXct(strptime(senser_df$Time,format='%Y-%m-%d %H'))
  senser_df$PM25 <- as.numeric(senser_df$PM25)
  senser_df$O3   <-as.numeric(senser_df$O3)
  # Data Cleaning
  senser_df <- senser_df[!senser_df$PM25 %in% c("ND","",NA), ]
  row.names(senser_df) <- c(1:length(senser_df[,1]))
  return(senser_df)
}
```

D.8 RNNmodel.R Code

```
### RNN model Functions
    --------------------------------------------------------------------------


PM25_pred<-function(senser_O3_pred,model,PM25_o){
  # PM25_o<-main_data_clean(senser_Fengyuan_db)$PM25_t
  # senser_O3_pred<-main_data_clean(senser_Fengyuan_db)$O3_t
  # model<-model_Fengyuan

  Omax <- max(PM25_o)
  Omin <- min(PM25_o)

  O3_o <- matrix((senser_O3_pred)/2, nrow = 5)
  PM25_p <- predict(model, O3_o[])
  PM25_pb <- PM25_p*(Omax-Omin)+Omin
  return(as.vector(t(PM25_pb)))
}

predict<-function(model, O3_t) {
  return(predictr(model, O3_t))
}

model<-function(X,Y,learning_rate,hidden_dim,num_epochs) {
  model <- trainr(Y = Y,
                  X = X,
                  learningrate = learning_rate,
                  hidden_dim = hidden_dim,
                  numepochs = num_epochs
                  #,
                  #loss_function = 1
  )



  return(model)
}
```

Appendix E

RNN Training Visualization Code

E.1 Server.R Code

```
library(RMySQL)
library(dplyr)
library(rnn)
library(reshape2)
library(plotly)
library(tictoc)
library(rlist)
conn <- dbConnect(MySQL(),dbname = "idw&ipdw", username="root",password="123", host
    ="140.128.101.187")
dbSendQuery(conn,"SET NAMES utf8")
rnn_1hr_db <- dbGetQuery(conn, "select * from opendata_hr_Dali")
rnn_1hr_db<-rnn_1hr_db[order(rnn_1hr_db$date),]

# db to df
rnn_o_df <- data.frame(PM25=rnn_1hr_db$PM25,O3=rnn_1hr_db$O3,Time=paste(
    rnn_1hr_db$date," ",rnn_1hr_db$hour,sep = ""))
# Time format
rnn_o_df$Time<-factor(rnn_o_df$Time)
rnn_o_df$Time<-as.POSIXct(strptime(rnn_o_df$Time,format='%Y/%m/%d %H'))

xx<-rnn_o_df[length(rnn_o_df),3]-as.POSIXct(Sys.time())
rnn_o_df[,3]<-rnn_o_df[,3]-xx

data_size<-function(num_data_size) {
  #data_size <- "400, 2800"
  data_size <- toString(num_data_size)
  data_size<-strsplit(data_size,split=", ",fixed=T)
  data_size_df<-data.frame(matrix(unlist(data_size), nrow=1, byrow=T))
  #class(data_size_df[,1])
  num_data_size_df<-data.frame(max_size=as.numeric(as.character(data_size_df[,2])),
    min_size=as.numeric(as.character(data_size_df[,1])))
  #min_size<-as.numeric(as.character(data_size_df[,1]))
  return(num_data_size_df)

}

predict<-function(model, O3_t) {
  return(predictr(model, O3_t))
}

model<-function(X,Y,learning_rate,hidden_dim,num_epochs) {
  tic()
  model <- trainr(Y = Y,
                  X = X,
                  learningrate = learning_rate,
                  hidden_dim = hidden_dim,
                  numepochs = num_epochs
  )
  ttime <- toc()
  ttime <- ttime$toc - ttime$tic3
  list.save(model, file="model.RData")

}

function(input, output) {

  output$plot_t <- renderPlotly({

    num_data_size_df<-data_size(input$num_data_size)

    input$lagtime
    #num_data_size<-input$num_data_size
    train_test <- (2571+num_data_size_df$min_size):(2570+num_data_size_df$max_size)#
    train_test <- 2871:2970
    # #rnn_t_df<-rnn_o_df[train_test,]
    train_test_move <- (2570+num_data_size_df$min_size+input$lagtime-4):(2569+
    num_data_size_df$max_size+input$lagtime-4)#train_test_move <- 2570:2969
    rnn_t_df_move <- data.frame(PM25=rnn_o_df[train_test,1],O3=rnn_o_df[
    train_test_move,2],Time=rnn_o_df[train_test,3])

    plot_ly(rnn_t_df_move, x = ~Time , y = ~O3, name = "O3", type = 'scatter', mode =
    'lines') %>%
      add_trace(y = ~PM25, name = "PM2.5", connectgaps = TRUE) %>%
      layout(title = "Original Data",xaxis=list(title = "",fixedrange=FALSE),yaxis=
    list(title = "Value",fixedrange=TRUE))
```

```
})

output$plot_p <- renderPlotly({

  num_data_size_df<-data_size(input$num_data_size)

  input$lagtime
  #num_data_size<-input$num_data_size
  train_test <- (num_data_size_df$min_size):(num_data_size_df$max_size)#train_test
  train_test_move <- (num_data_size_df$min_size+input$lagtime):(
  num_data_size_df$max_size+input$lagtime-4)#train_test_move

  rnn_t_df_move <- data.frame(PM25=rnn_o_df[train_test,1],O3=rnn_o_df[
  train_test_move,2],Time=rnn_o_df[train_test,3])
  O3_o <- matrix((rnn_t_df_move$O3)/2, nrow = 5)
  PM25_o <- matrix((rnn_t_df_move$PM25)/2, nrow = 5)

  Omax <- max(PM25_o)
  Omin <- min(PM25_o)

  # Standardize in the interval 0 - 1
  O3_t <- (O3_o - min(O3_o)) / (max(O3_o) - min(O3_o))
  PM25_t <- (PM25_o - min(PM25_o)) / (max(PM25_o) - min(PM25_o))

  ## Transpose
  O3_t <- t(O3_t)
  PM25_t <- t(PM25_t)

  # Training-testing sets
  tnum<-length(rnn_t_df_move[train_test,1])/5/5*4
  train <- 1:tnum
  test <- (tnum+1):(length(rnn_t_df_move[train_test,1])/5)
  time_test_range <-(length(rnn_t_df_move[train_test,1])/5*4+1):length(rnn_t_df_move
  [train_test,1])

  learning_rate = as.numeric(toString(input$learning_rate))
  hidden_dim = as.numeric(toString(input$hidden_dim))+1
  num_epochs = as.numeric(toString(input$num_epochs))+1

  # Train model. Keep out the last two sequences.
  tic()
  model(O3_t[train,],PM25_t[train,],learning_rate,hidden_dim,num_epochs)
  # model<-model(O3_t[train,],PM25_t[train,],learning_rate,hidden_dim,num_epochs)
  ttime <- toc()
  ttime <- ttime$toc - ttime$tic3

  #
  model = list.load("model.RData")
```

```
    # Predicted values
    PM25_p <- predict(model, O3_t)
    PM25_pb <- PM25_p*(Omax-Omin)+Omin

    # MAPE 1hr to 8hrs
    actual<-data.frame(as.vector(PM25_o[,test]))
    pred<-data.frame(as.vector(t(PM25_pb[test,])))
    mape<-mean(abs((actual[,1]-pred[,1])/actual[,1]))

    # print mape
    mape_str<-paste("MAPE = ", mape)

    # print train time
    ttime_str<-paste("Training time = ", ttime,"(sec)")

    # Plot predicted vs actual. Training set + testing set
    p1<-plot_ly(rnn_t_df_move, x = rnn_t_df_move[,3] , y = as.vector(PM25_o), name = "
    Actual PM2.5", type = 'scatter',fill = 'tozeroy',  mode = 'lines', fill = "tozeroy
    ",fillcolor = 'rgba(168, 23, 123, 0)') %>%
      add_trace(y = as.vector(t(PM25_pb)), name = "predicted PM2.5", connectgaps =
    TRUE , fill = "tonexty", fillcolor = 'rgba(190, 190, 190, 0.1)') %>%
      layout(title = "PM2.5 Actual vs predicted",xaxis=list(title = ttime_str,
    fixedrange=FALSE),yaxis=list(title = "Value",fixedrange=TRUE))

    # Plot predicted vs actual. Testing set only.
    p2<-plot_ly(rnn_t_df_move, x = rnn_t_df_move[time_test_range,3] , y = as.vector(
    PM25_o[,test]), name = "Actual PM2.5", type = 'scatter',fill = 'tozeroy',  mode =
    'lines', fill = "tozeroy",fillcolor = 'rgba(168, 23, 123, 0)') %>%
      add_trace(y = as.vector(t(PM25_pb[test,])), name = "predicted PM2.5",
    connectgaps = TRUE, fill = "tonexty", fillcolor = 'rgba(190, 190, 190, 0.1)') %>%
      layout(title = "PM2.5 Actual vs predicted",xaxis=list(title = mape_str,
    fixedrange=FALSE),yaxis=list(title = "Value",fixedrange=TRUE))

    p <- subplot(p1, p2, shareY = TRUE, titleX = TRUE )

  })

}
```

E.2 ui.R Code

```
library(shiny)
library(shinydashboard)
library(plotly)
```

```
actionLink <- function(inputId, ...) {
  tags$a(href='javascript:void',
         id=inputId,
         class='action-button',
         ...)
}

header <- dashboardHeader(
  title = "Recurrent Neural Networks Model Training (Taichung Dali)",# Model Training(
     台中大里)
  titleWidth = "100%"

)

body <- dashboardBody(
  includeCSS("css/bgcolor.css"),
  fluidPage(
    #titlePanel("Movie explorer"),
    fluidRow(
      column(3,
             wellPanel(
               sliderInput("num_data_size",
                           "Number of Data Size:",
                           min = 0,#1
                           max = 1600,
                           #value = 80
                           step = 100,
                           value = c(600,700),
                           round = TRUE
               ),

               sliderInput("lagtime", "Lagtime", 0, 24,
                           value = 4, step = 4 ),

               sliderInput("learning_rate", "Learning Rate", 0, 1,
                           value = 0.5, step = 0.1 ),

               sliderInput("num_epochs", "Number of Epochs", 0, 1000,#1
                           value = 100, step = 100),

               sliderInput("hidden_dim", "Hidden Layer", 0, 50,#1
                           value = 10, step = 5),

               # Include clarifying text ----
               #helpText("Notice: Lagtime=4"),

              submitButton("Let's Training!"),
              conditionalPanel(condition="$('html').hasClass('shiny-busy')",
                               tags$div("Loading...",id="loadmessage"))
             )
      ),
      column(9,
             plotlyOutput("plot_t"),
             plotlyOutput("plot_p")#,



      )
    )
  )

)#body end
dashboardPage(
  header,
  dashboardSidebar(disable = TRUE),
  body
)
```

E.3 custom.css Code

```
.plotly.html-widget.html-widget-output.shiny-bound-output.js-plotly-plot {
  z-index: 22;
  position: relative;
}

.plotlybars {
  padding: 0 10px;
  vertical-align: bottom;
  width: 100%;
  height: 100%;
  overflow: hidden;
  position: relative;
  box-sizing: border-box;
}

.plotlybars-wrapper {
  width: 165px;
  height: 100px;
  margin: 0 auto;
  left: 0;
  right: 0;
  position: absolute;
  z-index: 1;
}

.plotlybars-text {
  color: #447adb;
    font-family: 'Open Sans', verdana, arial, sans-serif;
  font-size: 80%;
  text-align: center;
  margin-top: 5px;
}

.plotlybars-bar {
  background-color: #447adb;
    height: 100%;
  width: 13.3%;
  position: absolute;

  -webkit-transform: translateZ(0);
  transform: translateZ(0);

  animation-duration: 2s;
  animation-iteration-count: infinite;
  animation-direction: normal;
  animation-timing-function: linear;

  -webkit-animation-duration: 2s;
  -webkit-animation-iteration-count: infinite;
  -webkit-animation-direction: normal;
  -webkit-animation-timing-function: linear;
}

.b1 { left: 0%; top: 88%; animation-name: b1; -webkit-animation-name: b1; }
.b2 { left: 14.3%; top: 76%; animation-name: b2; -webkit-animation-name: b2; }
.b3 { left: 28.6%; top: 16%; animation-name: b3; -webkit-animation-name: b3; }
.b4 { left: 42.9%; top: 40%; animation-name: b4; -webkit-animation-name: b4; }
.b5 { left: 57.2%; top: 26%; animation-name: b5; -webkit-animation-name: b5; }
.b6 { left: 71.5%; top: 67%; animation-name: b6; -webkit-animation-name: b6; }
.b7 { left: 85.8%; top: 89%; animation-name: b7; -webkit-animation-name: b7; }

@keyframes b1 { 0% { top: 88%; } 44% { top: 0%; } 94% { top: 100%; } 100% { top: 88%;
    } }
@-webkit-keyframes b1 { 0% { top: 88%; } 44% { top: 0%; } 94% { top: 100%; } 100% {
    top: 88%; } }

@keyframes b2 { 0% { top: 76%; } 38% { top: 0%; } 88% { top: 100%; } 100% { top: 76%;
    } }
@-webkit-keyframes b2 { 0% { top: 76%; } 38% { top: 0%; } 88% { top: 100%; } 100% {
    top: 76%; } }
```

```
@keyframes b3 { 0% { top: 16%; } 8% { top: 0%; } 58% { top: 100%; } 100% { top: 16%; }
     }
@-webkit-keyframes b3 { 0% { top: 16%; } 8% { top: 0%; } 58% { top: 100%; } 100% { top
    : 16%; } }

@keyframes b4 { 0% { top: 40%; } 20% { top: 0%; } 70% { top: 100%; } 100% { top: 40%;
    } }
@-webkit-keyframes b4 { 0% { top: 40%; } 20% { top: 0%; } 70% { top: 100%; } 100% {
    top: 40%; } }

@keyframes b5 { 0% { top: 26%; } 13% { top: 0%; } 63% { top: 100%; } 100% { top: 26%;
    } }
@-webkit-keyframes b5 { 0% { top: 26%; } 13% { top: 0%; } 63% { top: 100%; } 100% {
    top: 26%; } }

@keyframes b6 { 0% { top: 67%; } 33.5% { top: 0%; } 83% { top: 100%; } 100% { top:
    67%; } }
@-webkit-keyframes b6 { 0% { top: 67%; } 33.5% { top: 0%; } 83% { top: 100%; } 100% {
    top: 67%; } }

@keyframes b7 { 0% { top: 89%; } 44.5% { top: 0%; } 94.5% { top: 100%; } 100% { top:
    89%; } }
@-webkit-keyframes b7 { 0% { top: 89%; } 44.5% { top: 0%; } 94.5% { top: 100%; } 100%
    { top: 89%; } }
```