City, University of London

MSc Data Science

Project Report

2022

Use of Transfer Learning on Recurrent Neural Networks for prediction of Air Pollution

Researcher: Rajani Mohan Janipalli

Supervised by: Dr. Oleksandr Galkin

Submitted: 03-03-2023

Declaration

By submitting this work, I declare that this work is entirely my own except those parts duly identified and referenced in my submission. It complies with any specified word limits and the requirements and regulations detailed in the assessment instructions and any other relevant programme and module documentation. In submitting this work, I acknowledge that I have read and understood the regulations and code regarding academic misconduct, including that relating to plagiarism, as specified in the Programme Handbook. I also acknowledge that this work will be subject to a variety of checks for academic misconduct.

Abstract

Air pollution is a serious contemporary problem, which kills about seven million people each year. nine out of ten people around the globe inhale polluted air. One way to tackle this problem is by predicting air pollution and taking necessary precaution to reduce its impact.

This project studies if transfer learning technique can improve the air pollution forecasting ability of machine learning models. The report elaborates how this was done by applying transfer learning technique on Recurrent Neural Networks that forecast the pollution levels for one hour ahead based on the data of last 24 hours. To carry out the study, five-year air pollution data and meteorological data of five major cities in India were used. The results of the study were evaluated with commonly used performance metrics.

Keywords: Transfer learning, model performance, recurrent neural networks, timeseries forecasting, air pollution prediction.

Table of Contents

Chapter 1 Introduction and objectives 9
1.1 Background to the problem 9
1.2 Purpose and beneficiaries of the project 11
1.3 Objectives 12
1.4 Methods 12
1.5 Metrics 13
1.6 Work plan 13
1.7 Structure of the report 13
Chapter 2 Context 15
2.1 Brief history of air pollution forecasting 15
2.2 Current state of art in air pollution prediction 16
2.3 Air pollution prediction in India 19
2.4 Transfer learning in time series 20
2.5 Transfer learning for air pollution prediction 21
2.6 Conclusion 21
Chapter 3 Methods 23
3.1 Software & Hardware infrastructure 23
3.2 Data acquisition 23
3.3 Data loading, data creation and initial pre-processing 24
3.4 Exploratory data analysis and further data pre-processing 29
3.5 Timeseries data preparation 34
3.6 Algorithms used for modelling 35
3.6.1 GRU 36
3.6.2 Bidirectional GRU 37
3.6.3 LSTM 38
3.6.4 Bidirectional LSTM 38
3.7 Hyperparameter tuning 39
3.8 Training, tuning and testing of Models 39
3.9 Evaluation 41
3.10 Conclusion 42
Chapter 4 Results 43
4.1 Initial data pre-processing 43
4.2 Exploratory data analysis of air pollution data 50
4.3 Further data pre-processing 54
4.4 Exploratory data analysis of air pollution and meteorological data 58
4.5 Training, tuning, testing and creation of best models 59

4.5.1 LSTM 59
4.5.2 Bidirectional LSTM or BiLSTM 62
4.5.3 GRU 66
4.5.4 Bidirectional GRU or BiGRU 68
4.6 Evaluation of transfer learning 72
4.6.1 For LSTM 72
4.6.2 For Bidirectional LSTM or BiLSTM 75
4.6.3 For GRU 78
4.6.4 For Bidirectional GRU or BiGRU 81
4.7 Conclusion 84
Chapter 5 Discussion 86
5.1 Examination of results with respect to objectives of the current project 86
5.1.1 Data pre-processing 86
5.1.2 Exploratory data analysis 86
5.1.3 Modelling 87
5.1.4 Evaluation 87
5.2 Examination of results with respect to previous studies 88
5.2.1 Previous air pollution studies 88
5.2.2 Previous transfer learning studies 88
5.2.3 Own contribution of researcher 89
Chapter 6 Evaluation, Reflections, and Conclusions 90
6.1 Choice of Objectives 90
6.2 Literature review examined 90
6.3 Methods used 90
6.4 Reflection 90
6.5 Lessons learnt from the project 91
6.6 Future work 91
6.7 Conclusion 91
Glossary 92
References 93
Appendix - A 98
Plots of exploratory data analysis 98
Appendix – B 104
Code testing instructions 104
Appendix - C 105
Code 105

Appendix – D 106
Project proposal 106

List of Figures

Figure 1.6.1: Work plan 13
Figure 3.5.1: Rolling window creation. 35
Figure 3.6.1: Structure and working of RNNs (Lazzeri, Francesca, 2020, p.147). 36
Figure 3.6.2: Typical structure of GRU (Bekkar et al., 2021, p.7). 37
Figure 3.6.3: Structure of Bidirectional GRU (Chen, Jiang and Zhang, 2019, p.118532). 37
Figure 3.6.4: Structure of LSTM (Lazzeri, Francesca, 2020, p.148). 38
Figure 3.6.5: Structure of Bidirectional LSTM (Bekkar et al., 2021, p.7). 38
Figure 4.1.1: Boxplot of all columns of station_hr_df2. 43
Figure 4.1.2: Distribution plots of PM2.5 44
Figure 4.1.3: Distribution plots of PM10 44
Figure 4.1.4: Distribution plots of NO 45
Figure 4.1.5: Distribution of NO2 45
Figure 4.1.6: Distribution plots of NOx 46
Figure 4.1.7: Distribution plots of NH3 46
Figure 4.1.8: Distribution plot of CO 47
Figure 4.1.9: Distribution plots of SO2 47
Figure 4.1.10: Distribution plots of O3 48
Figure 4.2.1: Histogram plots of PM2.5. 50
Figure 4.2.2: Line plot of monthly average values of PM2.5 51
Figure 4.2.3: Line plot of monthly average values of PM2.5 with other pollutants. 51
Figure 4.2.4: Line plot of monthly average values of PM2.5 with PM10 52
Figure 4.2.5: Scatter plots of PM2.5 vs other pollutants. 53
Figure 4.2.6: Scatter plots of PM2.5 vs other pollutants. 53
Figure 4.3.1: Correlation matrix of Delhi_air_met_df (initial). 54
Figure 4.3.2: Correlation matrix of Delhi_air_met_df (final). 55
Figure 4.4.1: PM2.5 variation with Meteorological factors (Delhi). 58
Figure 4.4.2: PM2.5 variation with Meteorological factors (Delhi). 58
Figure 4.4.3:PM2.5 variation with WinddirDegree (Delhi). 59
Figure 4.5.1:Baseline LSTM. 62
Figure 4.5.2: Best LSTM. 62
Figure 4.5.3: Baseline BiLSTM. 65
Figure 4.5.4: Best BiLSTM. 65
Figure 4.5.5: Baseline GRU. 68
Figure 4.5.6: Best GRU. 68
Figure 4.5.7: Baseline BiGRU. 71

Figure 4.5.8: Best BiGRU. 71
Figure 4.6.1: Best LSTM – Bengaluru (left – with TL, right – no TL). 73
Figure 4.6.2: Best LSTM – Hyderabad (left – with TL, right – no TL) 73
Figure 4.6.3: Best LSTM – Jaipur (left – with TL, right – no TL). 74
Figure 4.6.4: Best LSTM – Mumbai (left – with TL, right – no TL) 75
Figure 4.6.5: Best BiLSTM – Bengaluru (left – with TL, right – no TL) 76
Figure 4.6.6: Best BiLSTM – Hyderabad (left – with TL, right – no TL) 76
Figure 4.6.7: Best BiLSTM – Jaipur (left – with TL, right – no TL). 77
Figure 4.6.8: Best BiLSTM – Mumbai (left – with TL, right – no TL) 78
Figure 4.6.9: Best GRU – Bengaluru (left – with TL, right – no TL) 79
Figure 4.6.10: Best GRU – Hyderabad (left – with TL, right – no TL) 79
Figure 4.6.11: Best GRU – Jaipur (left – with TL, right – no TL). 80
Figure 4.6.12: Best GRU – Mumbai (left – with TL, right – no TL) 81
Figure 4.6.13: Best BiGRU – Bengaluru (left – with TL, right – no TL) 82
Figure 4.6.14: Best BiGRU – Hyderabad (left – with TL, right – no TL) 82
Figure 4.6.15: Best BiGRU – Jaipur (left – with TL, right – no TL) 83
Figure 4.6.16: Best BiGRU – Mumbai (left – with TL, right – no TL) 84

Chapter 1 Introduction and objectives

This chapter will give a brief introduction to the research carried out by the author of this report, who will be here after called as the researcher of the current project. The chapter consists of seven sections.

1.1 Background to the problem

This section gives a background of the problem which the current project tries to address in a way, and shows the motivation to do the current project.

At the *Opening address at the First WHO Global Conference on Air Pollution and Health* (WHO, 2018, para 7-14) it was said that about 90% global population inhales air contaminated by vehicular and factory exhausts, and burning of crops and garbage, taking 7 million human lives per annum which was evitable. It was also mentioned in the conference that contaminated air is the root cause for about 33% mortalities related to cardiac and chest area ailments (*Opening address at the First WHO Global Conference on Air Pollution and Health*, WHO, 2018, para 10).

It was highlighted at the conference that people living in critical areas, such as healthcare facilities, educational institutes etc., can be considered more vulnerable to the impact of air pollution. Therefore, quality of air must be measured in these places. Some places are not able to get the required measurement, but it could be extended by using cheap sensors, and other solutions which are cost-effective. More effective laws and norms are required to implement these kinds of changes and bring in the necessary technology (*First Global Conference on Air Pollution and Health*, 2018, WHO, para 7).

Manisalidis *et al.* (2020) reported that the most frequent contaminants in the air are Particulates Matter (PM) and ozone at the level of ground, and greater subjection to these contaminants affects health in both smaller and larger spans of time. They added that the immediate consequences can include annoyances to sensory organs and issues to different parts of thoracic region. They further specified that the detrimental effects on the 'respiratory, reproductive, and neurological systems' are among the gradual repercussions (Manisalidis *et al.*, 2020, p.7). Manisalidis *et al.* observed that due to persistent and recurring nature, the contaminants can also result in cancer and fatalities in rare cases. They also mentioned that the contaminants also produce effects on the environment which include acid rains, 'haze, stratospheric ozone' depletion and climate change globally, which result in (1) damage to ecosystem and infrastructure, (2) reduction of atmospheric transparency, (3) increase 'UV radiation' to hazardous levels and (4) surge in global warming and greenhouse gases, putting all living organisms in threat of survival (Manisalidis *et al.*, 2020, p.9).

As per Fuller *et al.* (2022) increase in population, development of cities and industries, use of conventional fuels, and lack of control policies, have led to a 66% rise in mortalities caused by air pollution, in the last 20 years. They reported that out of the nine million deaths from pollution in 2019,

over six and a half million were attributed to air pollution, making it the greatest contributor (Fuller *et al.*, 2022, pp. 535-536). Fuller *et al.* also said that in terms of threats, contamination of air is still among the key causes of ailments and early mortality in the world. They added that pollution has made more influence on health than any man-made disasters or life-threatening diseases, and it is responsible for as many fatalities as due to smoking. They found that 'Southeast Asia, east Asia and South Asia' witnessed these fatalities more than any other regions around the world (Fuller *et al.*, 2022, pp.e536-e537). Fuller *et al.* further reported that during 2019, the worldwide 'economic welfare losses' owing to indoor and outdoor air pollution amounted to 61% of the world's 'economic output', according to a World Bank analysis utilising GBD 2019 statistics on the health consequences (World Bank, 2022 cited in Fuller *et al.*, 2022, p.e539). They added that 'East Asia and the Pacific regions, and south Asia' were particularly affected economically by air pollution, with losses there equaling 9.3 and 10.3 percent of GDP, respectively (World Bank, 2022 cited in Fuller *et al.*, 2022, p.e539).

A global assessment of mortalities because of PM2.5 in 2012 revealed that China and India were the two nations with the most early deaths with figures of 3.91 million and 2.46 million respectively (Vohra *et al.*, 2021, p.5).

Dey and Chowdhury (2022) published that India was noted as a significant hotspot of pollution in the 'global burden of disease (GBD)' report, with some of the highest rates of 'premature deaths per 100,000' people worldwide (Dey and Chowdhury, 2022, p.239). They also found that over 98% of Indians are presently subjected to yearly PM2.5 levels that are higher than the recommended range of 10 $\mu g/m^3$ (micrograms per cubic meter) set by the WHO (Dey and Chowdhury, 2022, p.239). They further highlighted that in India, it is predicted that subjection to and outdoor PM2.5 will rise as a result of 'global warming' (Chowdhury et al., 2018; Kumar et al., 2018, cited in Dey and Chowdhury, 2022).

Sweileh *et al.* (2018) showed observed that China was one of the top ten active countries that published research papers on pollution-related respiratory health between the years 1900 to 2017 (Sweileh *et al.*, 2018, p.7). They found it surprising that on the same topic, the number of research papers published by India was less than half of that done by China, that clearly shows that there is very poor attention towards research on air pollution in India, which paves way for the studies on increase in air pollution levels in India, implying to less focus on monitoring air quality in India (Sweileh *et al.*, 2018).

According to an article by the World Bank, over 50% of the world's population inhales air that is of very poor quality, worse than WHO recommended levels. Most of them reside in China and India, and the population exposed to that poor quality air in India is higher than that in China (*WDI - The global distribution of air pollution*, no date).

Gulia *et al.* (2022) published that in terms of air quality monitoring, India has established a 'Continuous ambient air quality monitoring system (CAAQMS)', under which it has 278 'real-time monitoring stations' covering 147 cities (Gulia *et al.*, 2022, p.4). They also said that India has also created 'System

of Air Quality and Weather Forecasting and Research (SAFAR)' forecasting grid to observe purness of air, that offers air quality warnings in advance (Gulia *et al.*, 2022, p.4).

As per Kanaujia *et al.* (2022), recently the 'Indian Institute of Tropical Meteorology (IITM), India Meteorological Department (IMD) and the National Centre for Medium-Range Weather Forecasting (NCMRWF) under the Ministry of Earth Sciences' developed a 'high-tech Air Quality Early Warning System' for predicting high levels of air pollution, which does the prediction on a Multi Petaflop supercomputer and gives alerts to take necessary action (Kanaujia *et al.*, 2022, p.15).

To summarize, despite being one among the most polluted countries across the globe, India has started to put the required amount of focus on addressing the serious problem of air pollution only recently and a lot more research in the domain of air pollution is to be done to tackle that problem. Especially research related to the prediction of PM2.5 needs to be expedited to diminish premature deaths due to it as emphasized by Khilnani and Tiwari (2018).

Hence, the current project tries to address this problem of air pollution in India in a way described in the next section of this chapter.

1.2 Purpose and beneficiaries of the project

The purpose of the project is to show an aspect of diversification of the focus area of research on air pollution in India. *The project would serve this purpose by trying to answer the following research question:*

"If a recurrent neural network model trained and tuned on data of a city with high variance in PM2.5 concentrations is retrained using transfer learning on data of cities with comparatively lesser variance in PM2.5 concentration, can it improve the performance of the model?"

The answer to this research question will inform if more attention to research on air pollution in locations with higher variance of PM2.5 levels helps in the forecast of PM2.5 levels.

The benefits of the project are specific to but not limited to the following:

- Continuous ambient air quality monitoring system (CAAQMS)
- Indian Institute of Tropical Meteorology (IITM)
- India Meteorological Department (IMD)
- National Centre for Medium-Range Weather Forecasting (NCMRWF)
- Central Pollution Control Board (CPCB)
- Any other institutions undertaking researching in the area of air pollution.

1.3 Objectives

To answer the research question and fulfil the purpose of the project, the project has to be carried out systematically with the accomplishment of numerous tasks. The approach taken by the researcher to carry out the project in such a way, is to divide the tasks into a set of objectives which need to be achieved to complete the project. The objectives are as follows:

Perform literature review – To review numerous studies carried out on air pollution prediction or forecast and understand the kind of data used to do the research, methods applied to complete the research and how the results of the research were evaluated, in those studies.

Acquire data – To identify the right data required for the project based on review of literature and the purpose of the project, search for the sources of data, collect the data from them and load them into the environment in which the project will be executed.

Data pre-processing – To find out techniques to clean the data based on literature review, as per the need of the data and given the purpose of the project to prepare the data to a form useable for analysis.

Exploratory data analysis – To recognize meaningful patterns among the data, try to understand the underlying relations among the data and obtain the right set or subset of data that would be suitable for model creation. To know the city for which the variance in PM2.5 level is higher than others.

Modelling – To identify the appropriate machine learning algorithms to model, based on literature review, exploratory data analysis and the purpose of the project. Then create, train, tune and test the models.

Evaluation – To find the appropriate evaluation metrics from the literature review to evaluate the models and then discuss the critical points of the results of the project. And finally, with all of this done, provide an answer to the research question.

1.4 Methods

A literature review was done for air pollution prediction or forecast in terms of PM pollutants, methods used for such studies in the past, contemporary studies and studies carried out in India.

Acquired data through technique of API extraction of data, from data repositories.

Performed data pre-processing using imputation methods to fill missing values, remove outliers and scale data.

Performed exploratory data analysis through histograms to understand the distribution of data, through line plots to identify patterns among the data, through scatter plots to understand the relations among the data and through correlation matrix to find correlations among features.

Created and developed models of recurrent neural networks through hyperparameter tuning, trained, validated and evaluated model performance for test data, to project the upcoming levels of PM2.5.

Examined if transfer learning can boost the model performance as raised in the research question.

1.5 Metrics

Since the estimation of the upcoming levels of PM2.5 is a regression task, the model performance was evaluated with metrics for regression models by the use of loss functions and then calculate the extent of errors of prediction through comparison between predicted values and actual values of PM2.5.

1.6 Work plan

To achieve the objectives of the project, a work plan was prepared to execute the project in a systematic and ordered manner such that the tasks performed to achieve each objective would create outputs that would be input for the next objective in the order. The work plan is explained diagrammatically in Figure 1.6.1.

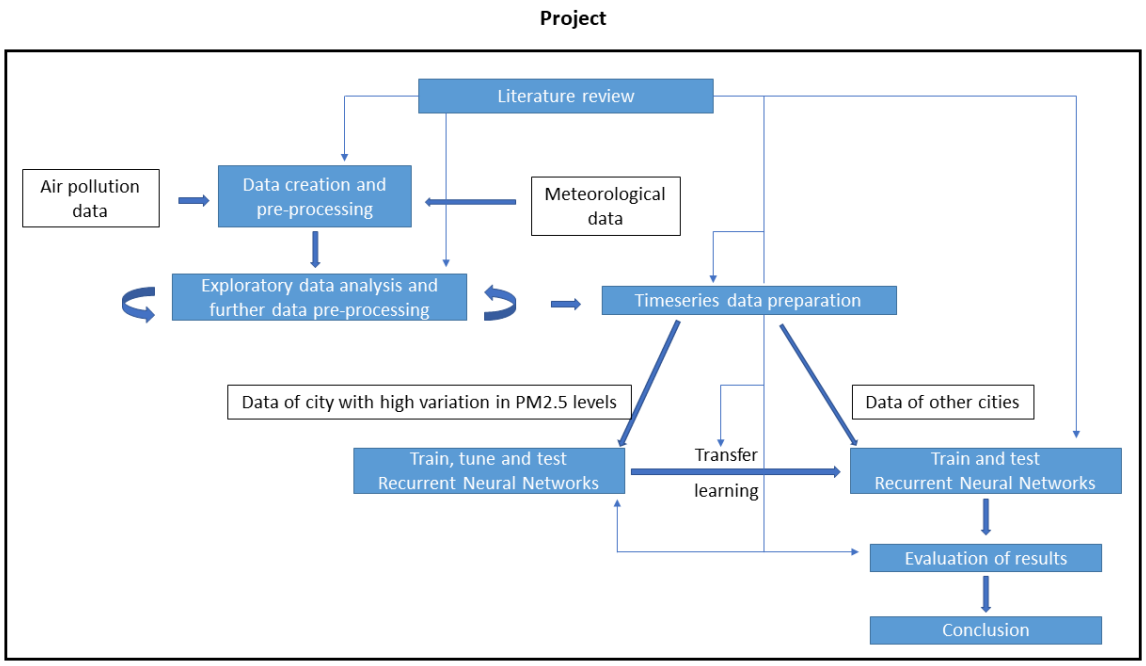


Figure 1.6.1: Work plan

1.7 Structure of the report

The report further is structured in different chapters as described below:

Chapter 2 – Context: this chapter includes highlights of understanding of studies in the literature review, summaries of the studies and takeaways from the studies.

Chapter 3 – Methods: this chapter explains the methods used to execute the current project.

Chapter 4 – Results: this chapter shows the results of various tasks performed to achieve the objectives of the current project.

Chapter 5 – Discussion: this chapter compares the results in terms of the extent to which they could achieve the objectives of the current project and also with the results of similar studies from the literature review.

Chapter 6 - Evaluation, Reflections, and Conclusions: this chapter evaluates the project from an overall point of view in aspects such as how different was the execution of the project from that of the plan, what are the limitations of the project, and what else could have been done and the scope of future work.

Chapter 2 Context

This chapter will give a brief summary of some previous studies published in the fields of air pollution prediction and transfer learning for prediction. The chapter consists of six sections.

2.1 Brief history of air pollution forecasting

This section gives a brief history of how air pollution prediction has evolved over the time.

Bai *et al.* (2018) reported that estimation of pollution in the air started in the 1960s as a result of advancements in studies and regularization of emissions. They said that 'numerical models, statistical models and potential forecasts' are the three categories of methods to project pollution (Bai *et al.*, 2018, p.3). Bai *et al.* mentioned 'potential forecasts' to be those built on the blend of the other two categories, and most popular in the 1960s. They explained that 'statistical forecasting' techniques examine the events without understanding the underlying cause of the change and these are further subdivided into 'parametric models' that consist of conventional models like 'principal component analysis and linear regression', and 'non-parametric models' consisting unspecified state like 'neural networks' (Bai *et al.*, 2018, p.10). Bai *et al.* informed that from the 1970s 'numerical models' became popular, then there were 'hybrid systems' that resulted from the amalgamation of 2 or additional models that maximise the benefits of each of them and increase projection accuracy (Bai *et al.*, 2018, pp.4-35).

McCollister and Wilson (1975) created linear stochastic models that employed time series data from the Los Angeles basin to estimate peak per day and per hour quantities of pollutants with inputs historical peak levels per day and per hour mean levels of pollutants. The comparable performance from the work of McCollister and Wilson advised that models using atmospheric data along with pollutant levels and utilizing pattern recognition approaches or multi-dimensional times series must be examined. Revlett (1978) constructed an empirical model to predict Ozone using pollutant concentrations and meteorological data and achieved errors of less than 25%.

Finz, Fronza and Spirito (1980) presented variable stochastic models with variable inputs to project everyday sulphur dioxide levels in Milan, Italy, throughout the warm season. They obtained a cyclic correlation from each model that, at the conclusion of every day, enabled anticipation of upcoming levels of pollution based on the present levels and climatic observations. Their projection accuracy was acceptable range (Finz, Fronza and Spirito, 1980).

From the early 1990s neural networks came into use for air pollution prediction, with research like that by Boznar, Lesjak and Mlakar (1993), who used neural networks to develop a method for short-term air pollution prediction using pollutant concentrations along with metrological conditions in Slovenia. Comparison among three multiple regression models and the ARIMA model for forecasting daily maximum ozone at Athens by Chaloulakou, Assimacopoulos and Lekkas (1999) showed that multiple regression models performed better than the ARIMA model.

Kolehmainen, Martikainen and Ruuskanen (2001) utilized recuring parts in data and neural networks for estimating quality of air in Stockholm. They compared a regression model with 'multi-layer perceptron, self-organizing maps, and combinations of periodic components' with both of the remaining algorithms, for their ability to estimate nitrogen dioxide from its historical per hour time series data and weather data (Kolehmainen, Martikainen and Ruuskanen, 2001, pp.815-819). They inferred from their results that the neural networks performed better and also that both pollutant time series data and meteorological data were necessary for modelling (Kolehmainen, Martikainen and Ruuskanen, 2001, p.824).

Kukkonen *et al.* (2003) examined 'deterministic modelling system, a linear statistical model and 5 neural networks models', for forecasting nitrogen dioxide and particulate matter, with inputs consisting of weather data and chosen traffic data, along with hourly pollutant concentrations of central Helsinki (Kukkonen *et al.*, 2003, pp.4539-4543). They concluded that NNs performed better than linear models as well as DET as the data itself was nonlinear (Kukkonen *et al.*, 2003).

Hooyberghs *et al.* (2005) created a model based on neural network to anticipate the per day particulate matter emissions in Belgium a day in advance, with PM10 levels and meteorological data for training and testing. One of the key conclusions of Hooyberghs *et al.* was the identification of the most important input parameter out of the meteorological data, excluding which the remaining parameters did not boost the model performance greatly on their inclusion as input to the model.

2.2 Current state of art in air pollution prediction

This section briefs the work of research done across the world outside India in recent years in the domain of air pollution prediction.

Biancofiore *et al.* (2017) employed 'Elman Recurrent Network (ANNE)' (Elman, 1990 cited in Biancofiore *et al.*, 2017, p. 653) to forecast PM2.5 and PM10 quantities for up to 3 days in advance. They utilized five weather data features and nine pollution data features for two years in Pescara, Italy. Their results for PM10 prediction informed that their proposed performed better than the other compared ones for a day and three days advanced predictions. For PM2.5 prediction, their proposed model performed better in all lengths of prediction (Biancofiore *et al.*, 2017, pp.653-657). The features in the data used by Biancofiore *et al.* are closely similar to the data considered for the current project. Inclusion of levels of organic compounds in those was an uncommon thing that Biancofiore *et al.* have experimented in their research, although reflecting the impact of this on their results would also have been helpful. Nonetheless, this is one of the literatures that aided data acquisition for the current project.

Sun and Sun (2017, pp. 145-147) employed 'Least square support vector machine (LSSVM)' after dimensionality reduction to estimate the per day levels of PM2.5. They used six pollution data features and two weather data features for one year in Baoding City in China. They assessed the relative

effectiveness of their proposed model in terms of 'root mean square error (RMSE), mean absolute percentage error (MAPE) and mean absolute error (MAE)' (Sun and Sun, 2017, p. 147). Brilliant idea of Sun and Sun to extract important features is valuable technique that was considered for the current project due to many features in the data. Their results supported that further as they improved model performance to a good range (Sun and Sun, 2017, p. 150). Also, the way they had split the data for training and validating models preserved chronology of timeseries data, another valuable learning for the current project. Further, two of the evaluation metrics used by Sun and Sun are commonly used and are considered for the current project.

Huang and Kuo (2018) projected PM2.5 with a model created by amalgamation of 'Long Short-Term Memory (LSTM)' and 'Convolutional Neural Network (CNN)', which they called 'APNet' (Huang and Kuo, 2018, p.1). They used two climate data features and just one pollution data feature for four years in Beijing, China (Huang and Kuo, 2018, p.9). It was clever of Huang and Kuo to put CNN that would extract features to boost the ability of RNNs, a notable inspiration for current project. Their experimentation with uncommon activation function, along with data scaling and early stopping regularization techniques implemented, and two of the model examination criteria used, are all some of the guidelines for current project.

Zhai and Chen (2018) projected per day mean values of PM2.5 by creating a multi-layered ensemble model consisting of five different regressors. They employed seven pollution data features and four climate data features for about three and half years in Beijing (Zhai and Chen, 2018, p. 646). They strategically employed chronologically first $4/5^{th}$ of data to train and leftover $1/5^{th}$ to validate the model (Zhai and Chen, 2018, p. 657). They also applied early stopping regularization over two of the regressors (Zhai and Chen, 2018, p.656). Intellectual method of Zhai and Chen to extract features through four techniques is an important lesson to experiment with feature selection in the current project. Their strategy to chronologically divide the data for training and testing is another important lesson learned, in addition to regularization through early stopping.

Delavar *et al.* (2019) investigated estimation of PM10 and PM2.5 levels by two neural network regressors and two traditional regressors, to find out the best model of them for data of Tehran. They analysed two pollution data features and five weather data features for ten years at numerous locations, for their study (Delavar *et al.*, 2019, pp. 4-6). Delavar *et al.* employed spline multinomial and Fourier series techniques to replace absent data (Delavar *et al.*, 2019, p. 7). It was a smart step by Delavar *et al.* to experiment with two uncommon interpolation techniques to replace absent data that greatly enhanced model performance as reported in their results and hence is significant inspiration for current project to interpolate absent data with suitable technique. Their analysis for data of numerous locations, based on location as well as time, showed a way to handle data for multiple locations in the current project, along with the benchmarks they used to validate their results.

Tao *et al.* (2019) built a hybrid model that began with CNN of single dimension followed by 'bidirectional gated recurrent unit' and ended with a dense neural network, that they called 'convolutional-based bidirectional gated recurrent unit (CBGRU)' to estimate levels of PM2.5 (Tao *et al.*, 2019, pp. 76692-76693). They carried their research with one pollution data feature and seven atmospheric data features for four years in Beijing (Tao *et al.*, 2019, p. 76691). In their thorough investigation Tao *et al.* did enormous experimentation with 'model structure, neuron numbers, momentum, optimizer, call backs, dropout, epochs and batch size' as well as many other parameters and from their findings it is visible that their hybrid model was clear winner (Tao *et al.*, 2019, pp. 76694-76697). This has informed numerous parameters to tune in the current project to obtain best performing models. Another leaning from Tao *et al.* is the strategy to plug absent values of data based the that of earlier timeframe. Although they considered just a single pollution feature, an elaboration of its impact on findings would have been informative.

Jeya and Sankari (2020) projected PM2.5 through 'bidirectional LSTM model' employing four climate data features and one pollution features for four years in Beijing. Through their pioneering analysis Jeya and Sankari showed advantage of learning target from features dual ways to push the estimation power of the model by their findings. This was considered to experiment and implement in the current project. Relative examination of their model with more than one other model was considered for the current project. Also, the trials they made to tune and improve their model was also inspired in the current project.

Xu and Yoneda (2021) devised a complex model of numerous LSTM and dense neural network layers to estimate PM2.5 with analysis done both chronologically as well as geographically (Xu and Yoneda, 2021, pp. 2578-2581). They employed six pollution features and six weather data feature for one at multiple locations in Beijing (Xu and Yoneda, 2021, pp. 2581-2582). First point learned from the comprehensive study of Xu and Yoneda was the idea to construct a model that would blend geographical as well as chronological factors to boost the contaminant predictability. However, they evaluated their results for each location separately which is another lesson for the current project. In addition to these, their methods of pre-processing and tuning models are also inspiration for the current project.

Bekkar *et al.* (2021) constructed hybrid models with combinations of convolutional layers, recurrent network layers and dense network to project levels of PM2.5. They carried out their study with six pollution features and seven climate data features for four years in Beijing (Bekkar *et al.*, 2021, p. 9). In their wide-ranging investigation, Bekkar *et al.* checked the relative performance of multiple recurrent models and hybrid models, which is a great guideline for the current project. Also, the way they experimented with tuning of all the different models is an inspiration for the current project. All the benchmarks used by Bekkar *et al.* to validate their models were considered for the current project.

Gu, Li and Meng (2022) devised a complex hybrid model that could extract and select features, before estimating PM2.5 values. They employed one pollution data feature and 17 weather data features for three years in Beijing (Gu, Li and Meng, 2022, p. 128). Through their ground-breaking work Gu, Li and Meng have proven through their findings that understanding how each feature influences the target enhances the ability of the model. Influenced by their work, selection of features was also focused in the current project and similar principles were tried with recurrent neural networks. Apart from that, their techniques for pre-processing data were also informative for the current project.

Liu *et al.* (2022) projected levels of pollution in air through a 'random forest model' with seven climate data features for one year in Zhangdian District, China (Liu *et al.*, 2022, p. 9). Primary lesson learned from the influential research of Liu *et al.* is the method used by them to prepare time series data suitable for prediction based on timeframe and something similar was tried in the current project. Secondary lesson learned from their research was to use two different functions to search for optimal hyperparameters while tuning the models.

Muthukumar *et al.* (2022) constructed complex model with convolutional layer and RNN to project PM2.5 values, geographically as well as on time basis. They utilized numerous features of pollution data and weather data along with graphical data for two years of Los Angeles (Muthukumar *et al.*, 2022, pp. 2-8). In their pioneering study showed a way to generate graphical features from the existing data that they proved to greatly enhance the estimation capability of the model through their results. This is valuable lesson to create features in the current project based on the analysis.

2.3 Air pollution prediction in India

This section briefs the work of research done in India in recent years in the domain of air pollution prediction.

Chelani, Gajghate and Hasan (2002) built a three-layer neural network to estimate PM10 and metallic pollution with atmospheric data for Jaipur city in India and from the results of their study they showed lower values of errors for the predictions. Their study highlighted about use of momentum to boost performance of model (Chelani, Gajghate and Hasan, 2002). The city in which Chelani, Gajghate and Hasan carried out their study is one of the cities included in the current project.

Kumar and Goyal (2011) projected contamination in air Delhi in India through two statistical regressors and hybrid combination of those two regressors. They employed four pollution data features and 14 climate data features for seven years to carry out their research. In their influential study it was determined that the hybrid model outscored the other models due to focus on correlations (Kumar and Goyal, 2011). They informed from their findings the importance of correlations among the features on the estimation ability of the models.

Abhilash *et al.* (2018) derived a model from 'ARIMA' algorithm to estimate levels of pollutants in air in Bengaluru in India (Abhilash *et al.*, 2018, pp. 413-415). They utilized three pollution data features from 14 locations for over three years to perform their study (Abhilash *et al.*, 2018, pp. 415-416). Through their results Abhilash *et al.* highlighted that static data improves the accuracy of prediction by the model. The city covered in their work is also a city in included in the current project.

Gopu, Panda and Nagwani (2021) forecasted values of contaminants in air in Hyderabad in India with 'ARIMA model' (Gopu, Panda and Nagwani, 2021, pp. 49-52). They employed 12 features of pollutant data for their analysis and estimate each contaminant distinctly. In their useful work, Gopu, Panda and Nagwani guided about feedforward technique to substitute absent values in the data. They also highlighted about the importance of deletion and creation of features from the existing ones as per the requirement of analysis (Gopu, Panda and Nagwani, 2021). The metropolitan selected by them is another place that is part of the current project.

Krishan *et al.* (2019) built 'LSTM model' to project levels of four different contaminants in air at Delhi, the capital city of India (Krishan *et al.*, 2019, pp. 903-904). They utilized 21 features of pollution data, atmospheric data and vehicular data for three years (Krishan *et al.*, 2019, pp. 902-903). The features utilized by Krishan *et al.* have the closest similarity to the features considered in the current project and hence the way they made use of them is an important guideline for the current project. The way they experimented with many parameters and hyperparameters to optimize and regularize their models is another guideline. Also, they validate their models with an uncommon metric in addition to the most common ones, which inspired to validate models through numerous metrics in the current project. The location used by them is another location of the current project.

2.4 Transfer learning in time series

This section briefs the work of research done on time series forecasting by applying transfer learning in financial domain.

He, Pang and Si (2019) suggested a fresh approach to train models while implementing 'transfer learning' with multiple datasets and a method to select the datasets depending on similarity among them (He, Pang and Si, 2019, pp.25-27). The utilized time series data of financial stocks to perform their experiments (He, Pang and Si, 2019, p. 30). Through their thorough investigation He, Pang and Si have proven with their findings that projection ability of models is enhanced with implementation of transfer learning. Their findings also implied that greater the number of datasets used for transfer learning, higher is the surge in model performance.

Ye and Dai (2021) derived a hybrid model from convolutional layers and implemented 'transfer learning' on it with two datasets that are not similar. The exercised two different techniques to calculate distance between two dissimilar datasets and in turn used that for selection datasets for

transfer learning (Ye and Dai, 2021, pp. 4-6). In their innovative experiment Ye and Dai showed that transfer learning can be used to transfer even model parameters as well as features extracted. Also, they proved through their results that even with some differences in datasets, transfer learning can effectively enhance models ability to predict.

Pal and Kar (2022) blended 'fuzzy learning' with 'transfer learning' to derive a new technique to estimate stock values (Pal and Kar, 2022, pp. 6944-6945). The pioneering research of Pal and Kar showed that with experimentation their proposed approach could make better estimates, even in cases where the target value is not present. A relative performance of their proposed approach with other different kind of models would have been more informative.

2.5 Transfer learning for air pollution prediction

This section briefs the work of research done on time series forecasting by applying transfer learning in air pollution domain.

Ma *et al.* (2019) devised an approach by implementing 'transfer learning' over 'bidirectional LSTM' model to project levels of PM2.5 (Ma *et al.*, 2019, pp. 2-4). They employed just one pollution data feature for their analysis for three years in Guangdong, China. They exercised 'rolling window' technique to arrange data suitable for time series projection (Ma *et al.*, 2019, pp. 4-5). The key lesson from the influential study of Ma *et al.* (2019) is that experimenting with time intervals while arranging data for time series projection determine the optimal interval. Also, they proved from their results that their approach enhances the estimation ability of the model.

A year later again, Ma *et al.* (2020) derived another approach by implementing 'transfer learning' over 'stacked bidirectional LSTM' to estimate the levels of three pollutants (Ma *et al.*, 2020, pp. 2-4). The primary lesson learned from the detailed work of Ma *et al.* (2020) is that experimentation with structure of recurrent network and projection time interval while implementing transfer learning can enhance the projection ability of models, particularly in cases where inadequate data is available for analysis.

Fong *et al.* (2020) applied 'transfer learning' over 'LSTM' to project the levels of contaminants in air at Macau (Fong *et al.*, 2020, pp. 3-4). In their innovative study Fong *et al.* have revealed that transfer learning can boost the estimation ability of model even when the target variable of the two data sets are different. They have also highlighted that their approach could increase the computational efficiency.

2.6 Conclusion

The above literature review has taught how the field of air pollution had evolved starting with mathematical models, followed by statistical model and reached to advanced neural networks.

It was observed that for statistical prediction of PM2.5 and PM10, ANNs are the most widely applied methods, which was also reported by Liao *et al.* (2021), due to benefits that they offer in terms of tuning the models to enhance the prediction by the models.

In most of the studies, meteorological data was used together with air pollutants concentration. The data required for the study can be acquired from Government authorities directly or from the online data published by them or it can be acquired from open-source data repositories like UCI Machine learning Repositories. Air pollution data and meteorological data may not be available together from a single source. So, it might be necessary for too many different datasets to obtain the required set of variables for air pollution prediction.

Missing date of a field using different methods in different studies although interpolation methods like linear interpolation, spline interpolation, and time-based interpolation were the most commonly used methods. In some of the papers, missing values were filled with forward or backward filling or they've been replaced with mean or median. Few studies mentioned about outlier detection and removal using the Z score method. Many of the papers scaled their data before feeding it to machine learning algorithms and most of them used normalization techniques to convert the values to a range between 0 and 1.

Most of the studies used LSTM, GRU along with their bidirectional versions and CNN-LSTM algorithms for forecasting air pollution. Most of the papers predicted future values of particulate matter (PM2.5 or PM10). The transfer learning technique was majorly used among datasets with good similarity.

Most of the researchers validated their models with metrics such as RMSE, MAE and R^2 which also happen to be the most commonly used evaluation metrics regression tasks.

Hence, all of these inferences from the literature review give a Bunch of guidelines that can help to carry out the current research project which involves the estimation of upcoming PM2.5 values and with this, the objective to perform a literature review was accomplished.

Chapter 3 Methods

3.1 Software & Hardware infrastructure

The current research project was carried out on google Colab pro and Tableau for exploratory data analysis (EDA). The specifications of the hardware and software infrastructure used for the project are detailed in Table 3.1.1.

Table 3.1.1: Hardware and Software infrastructure used.

Entity	***Specification***
Integrated development environment (IDE)	Google Colab Pro
CPU allocated	Intel(R) Xeon(R) CPU @ 2.00GHz
GPU allocated	Tesla T4
Free memory allocated	13 GB
Software programs	Python, version 3.7.14 Tableau Desktop, version 2021.3.6 64-bit
Python library packages	Tensorflow version 2.8.2 Keras version 2.8.0 Pandas version 1.3.5 Numpy version 1.21.6 Matplotlib version 3.2.2 Scikit learn version 1.0.2 Permetrics version 1.2.2

3.2 Data acquisition

Chelani, Gajghate and Hasan (2002) obtained data from the reports of the 'National Environmental Engineering Research Institute (NEERI), Nagpur, India', for their study in Jaipur City (Chelani, Gajghate and Hasan, 2002, p. 806). Kumar and Goyal (2011) collected data from 'Central Pollution Control Board (CPCB), Delhi' for their study in Delhi (Kumar and Goyal, 2011, p. 5518). Abhilash *et al.* (2018) acquired data from 14 monitoring stations for their study in Bengaluru City. Gopu, Panda and Nagwani (2021) used data shared by CPCB, from the Kaggle community website. Krishan *et al.* (2019) collected data from local authorities for their study in Delhi. For all these studies in India the researchers directly or indirectly relied on data from CPCB.

Given the location of the researcher of the current project is London, UK and other conditions, it was not possible to obtain data from local authorities in India. So, the alternatives available to acquire the data were to do an online search for the data shared by CPCB and other authorities, in open-source data repositories or scrape data from the official websites of the authorities.

As data were acquired from the UCI Machine learning repository by Tao *et al.* (2019), Jeya and Sankari (2020) and Bekkar *et al.* (2021), a search was done for both air pollution data and meteorological data in the same repository. But no suitable data sources were found. Then, through a search on the Kaggle community website, air quality data for different Indian cities between the years 2015 and 2020 (*Air Quality Data in India (2015 - 2020)*, no date a) and meteorological data for the eight most populous cities in India (*Historical Weather Data for Indian Cities*, no date a) between the years 2009 and 2019 were found. The air quality data is the data that is openly shared by CPCB, and the meteorological data is the one owned by worldweatheronline.com, extracted with the help of their API. An inspection of the air quality data sets on their web page revealed that the data suitable for the current project was available in two different CSV files out of five CSV files on the web page and rest of the three CSV files were unnecessary.

As per Acker and Kreisberg (2020, p.106) Application programming interfaces (API) are 'technologies of custody that enable the extraction and access to data'. The data sets were extracted from Kaggle web pages using Kaggle API, for which a registered account with Kaggle was required. A Kaggle API token was obtained from the registered account of the researcher, which was available for download as a JavaScript Object Notation (JSON) file. It was needed to upload the JSON file to the google drive of the researcher, as its access is needed to use Kaggle API to extract the datasets.

The first step towards data extraction from the Kaggle website using its API was to install the Kaggle python package on to google Colab notebook. Google drive of the researcher was mounted on to Colab notebook to provide access to the Kaggle API token JSON file. Then the air pollution data set and the meteorological data set were downloaded by implementing commands as mentioned on Kaggle public API documentation (*Public API Documentation*, no date). The data sets were downloaded as zip files on to the Colab instance storage, which were then unzipped to extract the CSV files of the data on to the Colab instance storage. Unzipping the air quality data extracted five CSV files and doing the same for meteorological data extracted eight CSV files. After that, the downloaded zip files and unnecessary CSV files related to air pollution data were deleted from the Colab instance storage.

3.3 Data loading, data creation and initial pre-processing

Before loading data, python libraries viz., pandas, matplotlib.pyplot and seaborn were imported as these are required for data loading, pre-processing and exploration. For better visibility of visualizations, the theme style of visualizations was changed from the default white background to dark style of the seaborn library.

The plan was to load air pollution data and meteorological data individually and then bring them to a condition where they can be merged.

First, the air pollution data from the "station_hour.csv" file was loaded into a pandas data frame and assigned a variable name called "station_hr_df". The data frame had 2589083 rows or observations and

16 columns or features, that contained hourly concentrations of pollutants. The details of the columns in this data frame are mentioned in Table 3.3.1.

Table 3.3.1: Details of variables in station_hour.

Feature name	***Description***	***Variable type***
StationID	Id of air quality monitoring station which recorded the observations of pollutant concentrations.	Categorical non-numeric
Datetime	Date and time (timestamp) at which the observation was recorded at a particular station.	Date-time
PM2.5	The concentration of PM2.5	Continuous numeric
PM10	The concentration of PM10	Continuous numeric
NO	The concentration of Nitrogen Oxide	Continuous numeric
NO2	The concentration of Nitrogen Dioxide	Continuous numeric
NOx	The concentration of Nitrous Oxides	Continuous numeric
NH3	Concentration of Ammonia	Continuous numeric
CO	The concentration of Carbon Monoxide	Continuous numeric
SO2	The concentration of Sulphur Dioxide	Continuous numeric
O3	Concentration of Ozone	Continuous numeric
Benzene	The concentration of Organic compound Benzene	Continuous numeric
Toluene	The concentration of the Organic compound Toluene	Continuous numeric
Xylene	The concentration of the Organic compound Toluene	Continuous numeric
AQI	The air quality index calculated from air pollutant concentrations	Continuous numeric

AQI Bucket	Air quality index buckets based on AQI value	Categorical non-numeric

The fraction of missing values for each of the columns of station_hr_df was checked and it was observed that the StationId and Datetime columns had no missing values. For the rest of the columns that fraction of missing values ranged from a minimum of 0.189568 for the NOx column to a maximum of 0.801482 for the Xylene column. Three of the pollutants in this data, viz., Benzene, Toluene and Xylene are organic compounds. In the literature review, it was observed that except for the study by Biancofiore *et al.* (2017), no other study included or had organic pollutants. Considering this factor, the columns of organic compounds were dropped from station_hr_df.

Next, air pollution monitoring stations data contained in the "staions.csv" file was loaded into a pandas data frame and assigned a variable name called "stations_df". The data frame had five columns/ features, viz., StationId, StationName, City, State and Status. The purpose of loading this data was to obtain the cities in which the air pollution monitoring stations in station_hr_df are located. For this, only the StationId and City columns were needed from stations_df and hence the remaining columns were dropped.

Then, both station_hr_df and station_df had StaionId as a common column and on its basis, stations_df was merged into stations_hr_df and the resultant data frame was assigned a variable name "station_hr_df1". As a result of dropping the columns of organic compounds and merging stations_df, stations_hr_df1 had 12 columns.

After that, the distinct cities in stations_hr_df1 for which the air pollution data was available were checked. These cities were compared with the cities for which meteorological data were available by checking the names of CSV files related to meteorological data available in Colab instance storage. It was realized that both air pollution data and meteorological data were available only for five cities viz., Bengaluru, Delhi, Hyderabad, Jaipur and Mumbai. So, the CSV files of meteorological data for other cities were deleted from Colab instance storage.

Hence the current project was carried out with air pollution data and meteorological data of five cities viz., Bengaluru, Delhi, Hyderabad, Jaipur and Mumbai.

Then station_hr_df1 was filtered to have only those observations for which the city value is one of the five cities. As a result, the number of observations in station_hr_df1 reduced to 1790565. Before proceeding further, the data types of all the columns of station_hr_df1 were checked and it was found that the data type of Datetime column was "object", but it must be datetime data type to utilize it for time series analysis. So, the data type of the column was changed with the pandas function and consequently, the data type of the column changed to datetime64.

Thereafter, a new column for the year of the observation was generated in station_hr_df1 from the Datetime column. The distinct values of the year column were checked and found to be ranging from 2015 to 2020. Also, for air pollution data the oldest timestamp of observations was found to be '2015-01-01 01:00:00' and the newest timestamp of observations was found to be '2020-07-01 00:00:00'.

Next, meteorological data of Bengaluru city from the 'bengaluru.csv' file was loaded into a pandas data frame and assigned a variable name called "bengaluru_met_df". The data frame had 96432 observations and 25 features that contained hourly meteorological data for Bengaluru city. The details of the columns in this data frame are mentioned in Table 3.3.2.

Table 3.3.2: Details of variables in Bengaluru_met_df.

Feature name	***Description***	***Variable type***
date_time	Date and time (timestamp) at which the observation was recorded.	Date-time
maxtempC	Maximum temperature in degrees Celsius.	Continuous numeric
mintempC	The minimum temperature in degrees Celsius.	Continuous numeric
totalSnow_cm	Snowfall intensity in centimetres.	Continuous numeric
sunHour	The number of hours for which there was sunlight.	Continuous numeric
uvIndex	An index factor of ultraviolet radiations.	Categorical non-numeric
uvIndex.1	An index factor of ultraviolet radiations.	Categorical non-numeric
moon_illumination	Illumination value of moonlight.	Continuous numeric
moonrise	Time of rising of Moon.	Time
moonset	Time of setting of Moon.	Time
sunrise	Time of rising of Sun.	Time
sunset	Time of setting of Sun.	Time
DewPointC	The temperature in degrees Celsius at which dew formation starts.	Continuous numeric
FeelsLikeC	The temperature in degree Celsius that was felt but not the actual.	Continuous numeric

HeatIndexC	An index factor of heat measured in degree Celsius.	Continuous numeric
WindChillC	Chillness due to wind measured in degree Celsius.	Continuous numeric
WindGustKmph	Speed of wind gusts in kilometres per hour.	Continuous numeric
cloudcover	A factor indicating the coverage of clouds.	Continuous numeric
humidity	Humidity value in percentage.	Continuous numeric
precipMM	Rainfall intensity in millimetres.	Continuous numeric
pressure	Pressure value in millibars.	Continuous numeric
tempC	The actual temperature in degrees Celsius	Continuous numeric
visibility	Distance or range of visibility	Continuous numeric
winddirDegree	The wind direction in degrees between 0 to 360.	Continuous numeric
windspeedKmph	Speed of winds in kilometres per hour.	Continuous numeric

On checking the data types of all the columns of bengaluru_met_df, it was found that the data type of date_time column was object type and hence it was converted to datetime64 data type as it was done earlier.

The next task was to drop the unnecessary columns from bengaluru_met_df. The columns uvIndex and unIndex.1 seemed to be repetitive and to find out if that is the case or not, the distinct values of both the columns and their counts were checked. While the uvIndex column consisted of numbers ranging from 4 to 8, the uvIndex.1 column consisted of numbers ranging from 1 to 10. Although both the columns represent similar meteorological factors, the data variance was higher in uvIndex.1 column. So, the uvIndex column with lower data variance was dropped. For hourly data, the columns maxtempC, mintempC and FeelsLikeC are redundant features for tempC, so they were also dropped. The columns moonrise, moonset, sunrise and sunset are meteorological factors that occur only once a day and they don’t fit well in the hourly time series analysis, consequently, they were dropped as well. Since snowfall

is a very rare phenomenon in the five Indian cities in which the current project is to be carried out, the totalSnow_cm column was also dropped. The decision to drop all these columns was inspired and supported by meteorological features used in studies by Biancofiore *et al.* (2017), Zhai and Chen (2018), Jeya and Sankari (2020). Even Tao *et al.* (2019) and Bekkar *et al.* (2021) also did not have any of those variables which were dropped from the current project, except snowfall, in the meteorological data of their studies.

Similar to station_hr_df1, a new column for the year of the observation was generated in bengaluru_met_df from the date_time column. With all the above-mentioned columns dropped and a year column added, the data frame then had 17 columns. The distinct values of the year were checked and found to be ranging from 2009 to 2020. Also, for meteorological data of Bengaluru city the oldest timestamp of observations was found to be '2009-01-01 00:00:00' and the newest timestamp of observations was found to be '2020-01-01 23:00:00'.

Next, the meteorological data of all the other four cities were loaded into pandas data frames and were assigned variable names called Bombay_met_df, delhi_met_df, Hyderabad_met_df and Jaipur_met_df. These four data frames had the same number of observations and features as bengaluru_met_df. All the data pre-processing actions performed on bengaluru_met_df till this point were also implemented on the meteorological data frames of all the other cities. The distinct values of the year of observations, oldest timestamp and newest timestamp were the same as that of bengaluru_met_df.

Comparing the distinct years and timestamps of the air pollution data frame and meteorological data frames of all five cities, it was observed that the data was consistently available for the five years from 2015 to 2019. So, all the data frames were filtered to have observations within this timeframe. For air pollution data, the filtered data frame was assigned a new variable name called station_hr_df2. As a result of filtering the years of the data sets, the observations in station_hr_df2 were reduced to 1502210 and those in the meteorological data frames of all the five cities were reduced to 43824.

3.4 Exploratory data analysis and further data pre-processing

The first step of exploratory data analysis was plotting a boxplot of all the columns of station_hr_df2, to understand the distribution of data in each of its columns in terms of quantiles and to identify the presence of outliers. The results of the boxplot are discussed in detail in chapter 4 of this report. Then histograms of all the columns of station_hr_df2 were plotted in the form of distribution plots to understand the type of distribution of data for each column. The mean and median were also marked in those plots to keep observation of change in the distribution of data after the imputation of missing values.

Inspired by the methods used by Jeya and Sankari (2020), Tao *et al.* (2019) and Gu, Li and Meng (2022) for filling missing values in their studies, it was decided to impute missing values based on values of

previous time stamps. To compare the changes in the distribution of data for all columns, before and after the imputation of missing values, a copy of station_hr_df2 was created and assigned a variable name station_hr_df2_copy1, in which the imputation of missing values was done. The imputation was done using the interpolate method of the data frame with time as the value of its method argument. But to perform this action the index column of the data frame must be the timestamp column. So, the index column of station_hr_df2_copy1 was changed to Datetime column before performing the imputation action. As a result of the action, all the missing values were filled except three values in the first rows of the data frame, which were then filled with the median values of the columns to which those three missing values belonged.

Then, distribution plots of all the columns of station_hr_df2_copy1 were plotted along with mean and median values indicated on them and were compared with those plotted earlier for station_hr_df2. As an additional confirmation, the descriptive statistics of all the columns of both data frames were also compared. The results of this comparative analysis are elaborated in chapter 4.

Before proceeding further, the values of station_hr_df2_copy1 data frame were sorted in chronological order of timestamps and the index column of the data frame was reset to serial number values. Then, the number of observations or rows in station_hr_df2_copy1 for each one of the five cities was checked and it was realized that the numbers were different for different cities, as shown in Table 3.4.1.

Table 3.4.1: Number of observations for each city in station_hr_df2_copy1.

City	***Number of observations***
Bengaluru	243874
Mumbai	88060
Delhi	925734
Hyderabad	183617
Jaipur	60925

After that, individual data frames of air pollution data were created for each one of the five cities from station_hr_df2_copy1, which were assigned variable names starting with respective city names, viz., Bengaluru_hr_df, Delhi_hr_df, Hyderabad_hr_df, Jaipur_hr_df, Mumbai_hr_df. These data frames had observations as mentioned in Table 3.4.1 corresponding to the city and 13 columns in all of them.

Since the data is an hourly time series, mathematically the expected number of observations for each city data frame can be calculated as:

$$expected\ no.of\ observations\ = no.of\ hours\ per\ day\ \times no.days\ in\ a\ year\ \times no.of\ years$$

$$= 24\ hrs\ \times 365\ days\ \times 5\ years$$

$$= 24\ hrs\ \times 365\ days\ \times 5\ years$$

$$= 43,800$$

In the five years of the observation period, one year must be a leap year, which means that additional 24 hours or 24 more observations. So, the expected number of observations for each city data frame is 43,824. Since the dimensions of the air pollution data frames of all the five cities need to be uniform, it was decided to do further analysis on just Bengaluru_hr_df and whatever data processing operations are performed on it, the same will be implemented on the data frames for other cities as well. The reason for specifically choosing Bengaluru_hr_df was it has a median value of observation out of all the five cities.

First, the counts of distinct values of StationId column were investigated, through which it was found that there were a different number of observations for different stations within the data of Bengaluru city. Then the sum of duplicate values of Bengaluru_hr_df as a whole was checked which resulted in 0 observations, whereas the sum of duplicate values in StationId column of Bengaluru_hr_df resulted in 243864 observations which is close to the total number of observations in that data frame. Similarly, there were 200051 duplicate observations in the Datetime column. Subtracting the number of duplicate observations in Datetime column (200051) from the total number of observations in the data frame (243874) results in 43823 observations, which is just one observation less than the expected number of observations calculated above. Looking at all these inferences collectively, it was concluded that there were duplicate observations for the same timestamp values because of the presence of observations for different stations during those values of timestamps.

In their study Xu and Yoneda (2021) considered the spatial aspect of different studies, but they didn't experiment with anything to combine the observations from different stations and made models for each station. As a novelty and experimentation, the researcher decided to average the values of air pollutants concentrations of different stations for the same timestamp value. This was done using the pivot table method of Bengaluru_hr_df data frame. The same process of pivot table was implemented for Delhi_hr_df, Hyderabad_hr_df, Jaipur_hr_df and Mumbai_hr_df. Since the averaging was done among different stations, the StationId column was eliminated. After this, the number of observations in the data frames of each city was as mentioned in Table 3.4.2.

Table 3.4.2: Number of observations of air pollution data frames of each city.

Data frame	***Number of observations***
Bengaluru_hr_df	43823
Mumbai_hr_df	43823
Delhi_hr_df	43823
Hyderabad_hr_df	43738
Jaipur_hr_df	22336

From Table 3.4.2, Bengaluru, Mumbai and Delhi have the same number of observations, whereas Hyderabad has a slightly fewer number of observations. Jaipur on other hand has all most half the number of observations as that of other cities. So, Jaipur would be significant in observing the impact of transfer learning in the context of study by Ma *et al.* (2020).

For further exploratory data analysis, the five data frames mentioned in Table 3.4.2 were concatenated vertically and the concatenated data frame was assigned a variable name airpoll_concat_df, which was then exported as a CSV file. That file was then imported as a data source in Tableau software, where exploratory data analysis of just air pollution data was done.

Firstly, histograms were plotted for the number of observations, average values of PM2.5, the minimum value of PM2.5 and the maximum value of PM2.5 for each city, to identify the city with the highest variance in PM2.5 concentrations. A line plot of per month mean values of PM2.5 was plotted to understand the presence of any periodicity and seasonality. A similar plot was plotted along with the monthly average values of other pollutants to understand the presence of any periodic and seasoned relation of PM2.5 with other pollutants. Then, scatter plots of PM2.5 against each of all the other pollutants were plotted to check if there is any formation of clusters for different cities. The results of all this exploratory data analysis are detailed in chapter 4.

After this, the data frames in Table 3.4.2 for the five cities were to be merged with their respective meteorological data frames. For this, some pre-processing was done in bengaluru_met_df, bombay_met_df, delhi_met_df, hyderabad_met_df and jaipur_met_df. The pre-processing included dropping the year column, renaming date_time column to Datetime and sorting the observations in chronological order of timestamps.

Exploratory data analysis of air pollution data concluded that Delhi city had the highest variance in PM2.5 concentrations. Hence, it was decided to use the data of Delhi city for further analysis and pre-processing and whatever actions are taken on the data frames of Delhi city, the same will be implemented to the data frames of other cities as well. This is in line with the basis for answering the research question of the current project.

As a part of further data creation, the meteorological data and the air pollution data of Delhi city were combined by merging delhi_met_df and Delhi_hr_df data frames respectively, based on timestamps since both the data frames had Datetime column in common. A new data frame was created by merging the two data frames and assigned a variable name called Delhi_air_met_df, which had 43,823 observations or rows and 25 features or columns that contained both meteorological data as well as air pollution data of Delhi city. To ensure that the data frame is in the correct order for a time series analysis, the observations of the data frame were sorted to be in chronological order of timestamps.

Inspired by the feature selection approach used by Bekkar *et al.* (2021), a feature selection was done based on correlation coefficients or correlation values among the features of Delhi_air_met_df. Before the creation of a correlation matrix that indicates the correlation values, it was checked that except for the Datetime column all the other columns were of either int or float data types to ensure that all of them are numeric data so that correlation values among them can be calculated. The correlation matrix was created by using the corr (correlation) method of Delhi_air_met_df and then a heatmap visual of the correlation matrix was plotted to see the correlation values among the columns. The results of the visual analysis of the correlation matrix heatmap are detailed in chapter 4. Based on those results, three columns viz., WindChillC, WindGustKmph and moon_illumination were dropped. So, the number of features or columns in Delhi_air_met_df was reduced to 22.

After this, the columns of Delhi_air_met_df were rearranged to have the PM2.5 column at the end, so that it can be easily assigned a target for prediction for modelling and also for time series data preparation. Unlike a frequent procedure used for identifying outliers by choosing a region that extends between thrice the standard deviation above and below the mean, Leys *et al.* (2013) used the median in place of the mean. Taking inspiration from Leys *et al.* (2013) to identify the outliers, first the descriptive statistics of all the columns of the data frame were checked and then the median, the 99 percentile and thrice the standard deviation of all the columns were evaluated. The evaluation showed that for some columns 99 percentile of the data was above thrice the standard deviation while for others it wasn't, due to which a common criterion for removal of outliers couldn't be set. Also, it was known that the sources of data were reliable and the possibility of presence of erroneously high or low values was very less. So, it was decided not to perform any outlier removal.

Similar to Delhi_air_met_df, merged data frames of meteorological data and air pollution data for the remaining four cities were also created and all the pre-processing done to Delhi_air_met_df was done on the data frames of other cities as well. The data frames were assigned variable names based on city name i.e., Bengaluru_air_met_df, Hyderabad_air_met_df, Jaipur_air_met_df and Mumbai_air_met_df. Finally, all five data frames were exported as CSV files to the google drive of the researcher, so that they can be imported into individual modelling Colab notebooks of different Recurrent Neural Network (RNN) algorithms.

Before going for modelling, further exploratory data analysis was done to identify any patterns and relations between PM2.5 and meteorological factors, the results of which are described in chapter 4.

3.5 Timeseries data preparation

Before feeding the combined air pollution and meteorological data to the RNNs, the data must be prepared and transformed into a format that can be used as input by models, to forecast a future target value. The process of time series data preparation was performed in the Colab notebook of each RNN and was done first directly on Delhi_air_met_df and then on the other five data frames using a researcher-defined function.

The first step under the time series data preparation was the creation of training and test datasets, in which the models would be trained and tuned on the training data; test data won't be exposed to the models during the training. Inspired by the approach taken by Sun and Sun (2017), Liu *et al.* (2022), Jeya and Sankari (2020) and Krishan *et al.* (2019) the first 80% of the observations in chronological order of Delhi_air_met_df were taken as the training dataset and leftover 20% as the test dataset so that there is a good amount of data to train the models and simultaneously the chronology of the time series is also preserved through this approach.

The second step was to scale the data in both training and test datasets. Following the technique applied by Huang and Kuo (2018), Bekkar *et al.* (2021) and Krishan *et al.* (2019), the datasets were scaled through normalization by using a Min-Max scaler to convert all the data into values in the range 0 to 1. In addition, the concept used by Tao *et al.* (2019) was also used and only the training dataset was fitted to the scaler object and then transformed, whereas the test dataset was just transformed but not fitted to the scaler object. It means that the data was scaled based on the training data. The number of observations in the training data was 35058 and that in the test data was 8765, whereas the number of columns in both sets was 21. For the models, all 21 columns were made as input features and only the PM2.5 column was made the target, making it a supervised learning.

The final step was to prepare time window samples of the data set. Inspired by the technique implemented by Ma *et al.* (2020), the Rolling method was applied to create samples of consecutive 24 hours data as observations. This means that each observation is a window of consecutive 24 hours data of all features after the rolling window creation. This is done as explained in Figure 3.5.1.

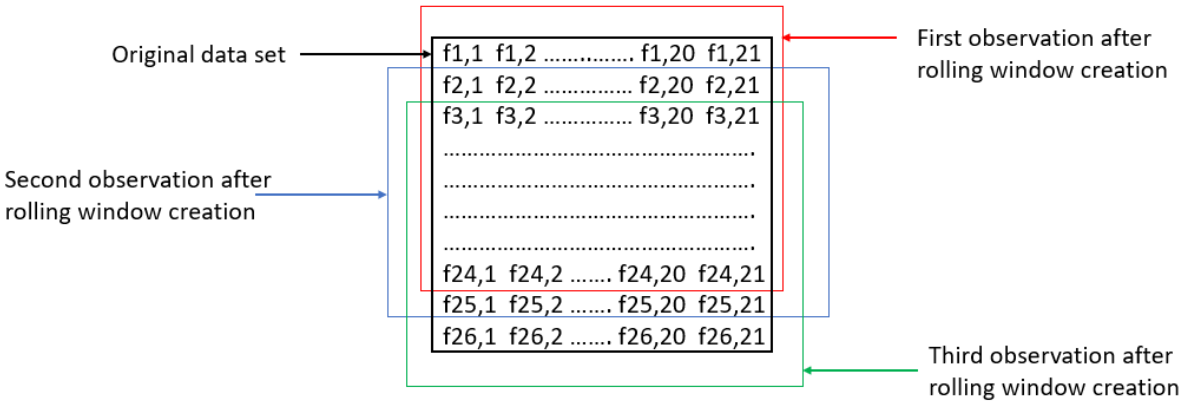


Figure 3.5.1: Rolling window creation.

If the dataset originally consists of 26 observations, for example, and 21 features, which is a 2D array, as depicted by a black box in Figure 3.5.1. So, each observation was a 1D array. After Rolling window creation, the first observation was a 2D array of the 1st 24 observations and 21 features of the original dataset as depicted by a red box in the figure, the second observation was a 2D array of 24 observations from the 2nd row to 25th row and 21 features of the original dataset as depicted by a blue box in the figure and the third observation was a 2D array of 24 observation from 3rdnd row to 26th row and 21 features of the original dataset as depicted by a green box in the figure. The target value for the first observation was the PM2.5 value of the 25th row in the primary dataset, for the second observation it was the PM2.5 value of the 26th row in the primary dataset and so on.

The dimensions of training and test datasets after rolling window creation are as detailed in Table 3.5.1.

Table 3.5.1: Dimensions of training and test datasets after rolling window creation.

Dataset (variable assigned)	***Dimension (observation, timesteps, features)***
Training features (X_train)	(35034, 24, 21)
Training target (y_train)	(35034,)
Test features (X_test)	(8741, 24, 21)
Test target (y_test)	(8741,)

To simplify, the models would take the data of all features for every consecutive 24 hours as input and try to project PM2.5 level for the subsequent hour.

3.6 Algorithms used for modelling

Four recurrent neural networks (RNNs) viz., Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), Gated Recurrent Unit (GRU) and Bidirectional GRU (BiGRU), were used for modelling in the current project, as they would be a good basis to answer the research question. Although many

studies in the literature review have included hybrid models like CNN-LSTM, the researcher decided to stick to the regular RNNs for a balanced comparison. Below is a brief description of all four RNNs.

Unlike other feedforward neural networks, RNNs have hidden cell stats that pass information in the memory from one node to the next node in different time frames (Lazzeri, Francesca, 2020). A typical structure and working of RNNs is as represented in Figure 3.6.1.

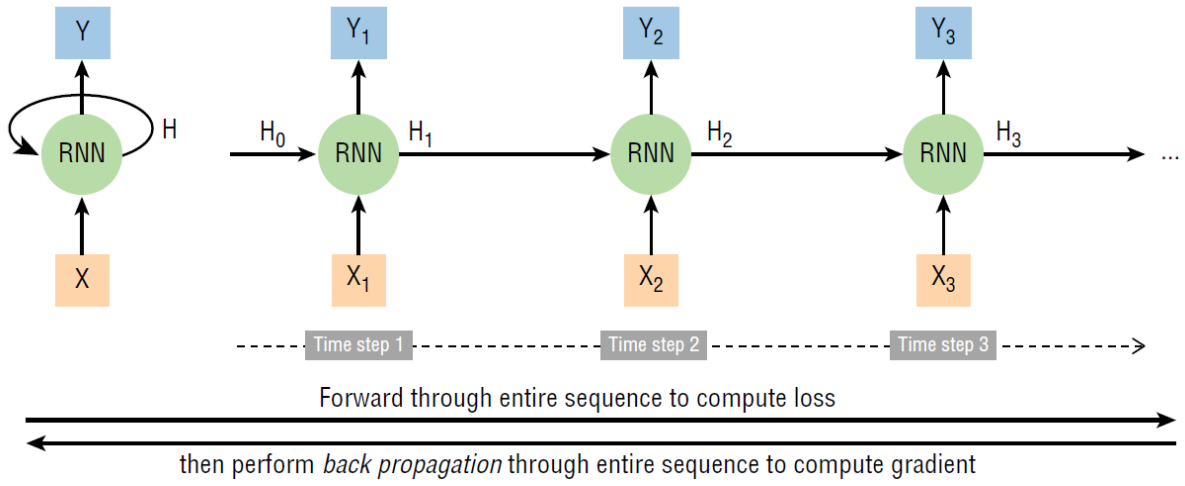


Figure 3.6.1: Structure and working of RNNs (Lazzeri, Francesca, 2020, p.147).

As shown in Figure 3.6.1, RNNs take the inputs from the current time step that is represented by X, information from the previous node that is represented by H and produces output that is represented by Y. The RNNs nodes shown are the same in different time frames. The information carried from the previous node is essentially about weights of X, Y and H, all passed on by H. As the length of the time sequence increases, during the back propagation, the gradients of the loss function are lost from the memory of the hidden cell, which is known as the problem of vanishing gradients and hence the weights are not updated to optimal levels (Lazzeri, Francesca, 2020, pp.145-147).

3.6.1 GRU

In the GRUs, the problem of vanishing gradients is solved with the help of a reset gate and an update gate inside its hidden state (Lazzeri, Francesca, 2020). A typical structure of GRU is depicted in Figure 3.6.2.

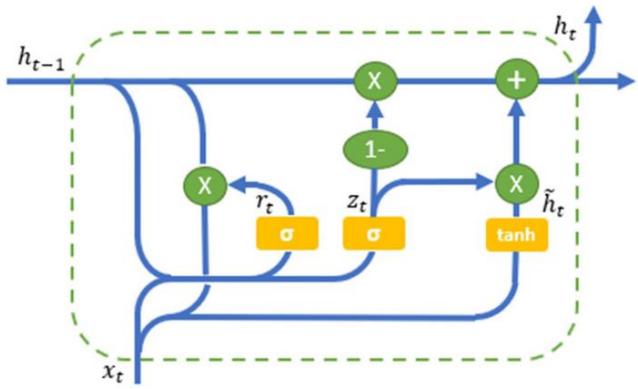


Figure 3.6.2: Typical structure of GRU (Bekkar et al.*, 2021, p.7).*

As shown in Figure 3.6.2, the hidden state of a GRU consists of a reset gate represented by the left-most X symbol and an update gate represented by the middle X symbol (Bekkar *et al.*, 2021). The decision of how much information from the past node output is to be forgotten is taken by the reset gate. Whereas, the decision of how much information from the past node is to be passed to the future is taken by the update gate. The sigmoid functions denoted by the sigma (σ) symbol help make the decisions by converting the weightage of information to values between 0 and 1, to produce probabilities of passing through the gates (Lazzeri, Francesca, 2020; Bekkar *et al.*, 2021).

3.6.2 Bidirectional GRU

Bidirectional GRU (or BiGRU hereafter) is an extension of GRU, a second hidden state is introduced such that the information is passed from the future to the present, which is the reverse of what happens in the GRU hidden state (Chen, Jiang and Zhang, 2019). A typical structure of Bidirectional GRU is displayed in Figure 3.6.3.

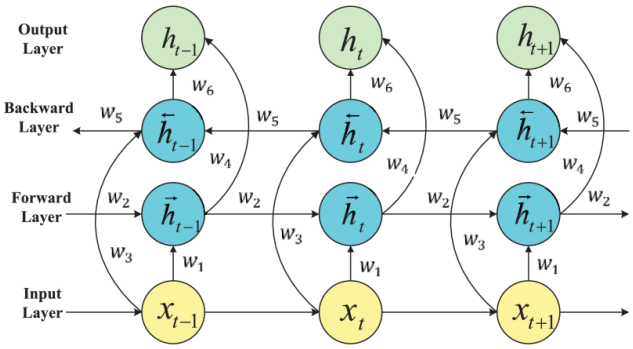


Figure 3.6.3: Structure of Bidirectional GRU (Chen, Jiang and Zhang, 2019, p.118532).

As shown in Figure 3.6.3, the information from past to present is passed through the forward layer hidden state and the information from the future to present is passed through the backward layer hidden state (Chen, Jiang and Zhang, 2019).

3.6.3 LSTM

LSTM has a cell state that stores information for a longer time dependency, in addition to an input gate, an output gate and a forget gate (Lazzeri, Francesca, 2020). A typical structure of LSTM is represented in Figure 3.6.4.

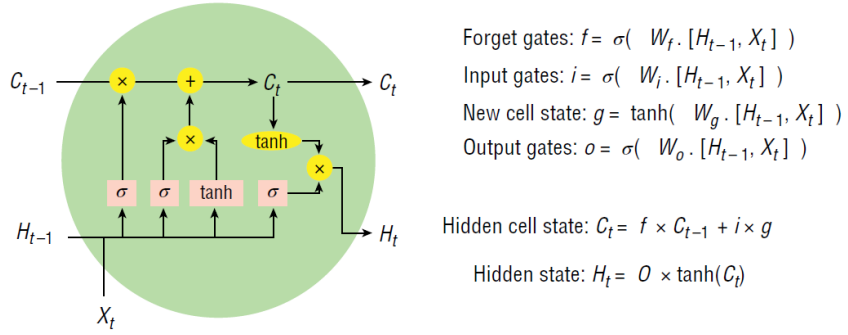


Figure 3.6.4: Structure of LSTM (Lazzeri, Francesca, 2020, p.148).

As shown in Figure 3.6.4, the cell state represented by a black straight arrow between C_{t-1} and C_t carries the information for long-term periods. The forget gate represented by left most X symbol, takes information from the earlier timestep (H_{t-1}) and present input (X_t) passes through a sigmoid function that assigns them weights between 0 and 1, to decide which information to forget. Similarly, the information to be passed to the cell state for long-term memory is determined by input gate. A speciality of LSTM is that it keeps on adding information to the past information in the cell state with an addition operation, represented by the plus (+) symbol. This summate information is carried forward for the long term. The next hidden state would be determined by output gate (Lazzeri, Francesca, 2020, pp.147-148).

3.6.4 Bidirectional LSTM

Bidirectional LSTM (or BiLSTM hereafter) has an additional hidden layer that passes information from future to present timestep (Bekkar *et al.*, 2021). A typical structure of Bidirectional LSTM is depicted in Figure 3.6.5.

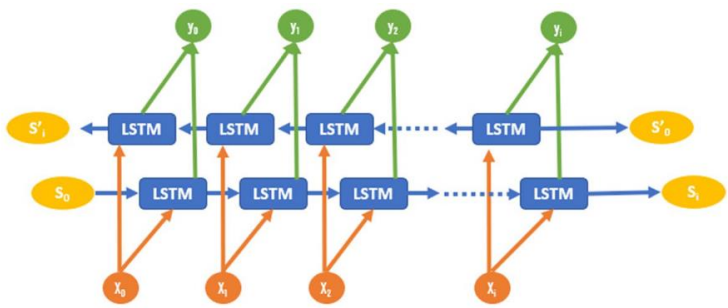


Figure 3.6.5: Structure of Bidirectional LSTM (Bekkar et al.*, 2021, p.7).*

As shown in Figure 3.6.5, the first hidden layer of the Bidirectional LSTM layer passes information from past to present timestep, whereas the second hidden layer passes information from future to present timestep. So, the model is trained by information in both directions (Jeya and Sankari, 2020).

3.7 Hyperparameter tuning

Unlike Krishan *et al.* (2019) and Liu *et al.* (2022) who employed grid search technique and random search method for hyperparameter tuning, the researcher of the current project has decided to use Hyperband and Bayesian-optimization methods for hyperparameter tuning following the suggestion of the supervisor.

One of the reasons for this it to bring some novelty to the project. Another reason is that as per Victoria and Maragatham (2021) grid search is difficult to expand to larger dimensions which leads to high computational cost, whereas random search although better than grid search but resorts to 'local optima' instead of reaching 'global optima' (Victoria and Maragatham, 2021, p.219).

Victoria and Maragatham (2021) also said that Bayesian-optimization incorporates previous knowledge about loss function and modifies consequent knowledge, helping to reduce the error and to boost the accuracy of the model (Victoria and Maragatham, 2021, p.219). As per Bertrand *et al.* (2017) Bayesian optimization attempts to represent the 'hyper-parameter space' as effectively as feasible for choosing 'next model to train' (Bertrand *et al.*, 2017, p.1).

Bertrand *et al.* (2017) also said that Hyperband tries to select set of models equally in defined model functions, which are then trained for a particular duration, followed by elimination of the poorest performers, and this process runs in cycle up to the point where limited choices are left. They then cautioned that it may also prematurely exclude some promising models (Bertrand *et al.*, 2017, p.3).

3.8 Training, tuning and testing of Models

The process of training, tuning and testing models for all four algorithms viz., LSTM, BiLSTM, GRU and BiGRU, has been the same. In all the cases, it was started with a baseline model and then a series of hyperparameter tuning was done to obtain the best model with the best hyperparameters. In all the cases this was done with Delhi_air_met_df, as Delhi city had a higher variance in PM2.5 level than other cities. The best model for each algorithm, trained and tested on Delhi_air_met_df was saved and then used for modelling with transfer learning on the data of other cities.

Before starting the baseline model, an early stopping criterion was created based on inspiration from Huang and Kuo (2018), Tao *et al.* (2019) and Zhai and Chen (2018). Similar to Tao *et al.* (2019), the patience argument of Keras early stopping function was set to 10, meaning that the training of the model would be stopped after no decrease in loss function value of validation set for 10 epochs and would effectively avoid overfitting of the model. 12.5 % of training data was assigned as a validation split, which mathematically approximates 10% of the total data. Since the time series data preparation was

done in such a way that each observation is a 2D array of consecutive 24 hours data, the chronology of the time series would be retained even in the validation split.

The baseline model was created with 64 neurons in a single RNN layer, which was followed by a dense layer with a single neuron. Mean Absolute Error (MAE) was used as the loss function and Adam was used as the optimizer for weight optimization. Then the model was fitted to X_train and y_train, with 100 epochs, batch size of 64, validation split of 0.125 (12.5%) and early stopping criteria, to train it. After the training of the model was completed, loss curves of training and validation sets were plotted, to compare it with that of the best model, as detailed in chapter 4. Then, the target values of PM2.5 were predicted with the model for X_test. Just as He, Pang and Si (2019) did, inverse normalization was done on the predict values of PM2.5 for test data y_pred, as well as on the actual values of PM2.5 for test data y_test, so that both can be compared in the original scale of the data. The model performance on test data was evaluated with an RMSE score. All this process was done to try multiple models with different values of best parameters obtained from hyperparameter tuning.

Performing hyperparameter tuning on the whole training data set would be computationally very expensive. So, a training set for hyperparameter tuning was created with the first 10,000 observations of X_train and y_train and a validation set for hyperparameter tuning was created with the next 2000 observations of X_train and y_train. Every time a hyperparameter tuning was to be done, a model building function was created with choices or a range of searches for hyperparameters, as it was needed as an argument in the Keras Hyperband function and Keras Bayesian-optimization function. For every set of hyperparameter tuning, the best parameters were searched using both Hyperband and Bayesian-optimization.

The hyperparameter tuning was first done for the number of neurons in a single layer RNN and then for the number of layers as well as number of neurons in a multilayer RNN. These two searches were carried out in multiple stages and with each progressive stage, the search for the best parameter was narrowed down. After the best configuration of neurons and layers was concluded, different models with that architecture were created with different activation functions, to identify the best activation function. Hyperparameter tuning was not done to search for the best activation function since it would be computationally more costly. Then the hyperparameter tuning was done for the best value for the dropout regularization parameter. As a novelty to the project as suggested by the supervisor, after all the hyperparameter tuning till this point, the weight optimizer was changed from Adam to AdamW, which is essentially Adam with a weight decay parameter. This would be an additional step of model regularization, and to the best of the knowledge of the researcher, it was not done by any of the studies in literature related to air pollution prediction or the use of transfer learning for time series. So, the final set of hyperparameter tuning was for the best value of weight decay.

After all the hyperparameter tuning was done as described above, models were created with all the best parameters and checked for the best accuracy for different batch sizes.

Finally, the best model was created with all the best parameters obtained through tuning and fitted to X_train and y_train with the best batch size obtained, 100 epochs, validation split of 0.125 and early stopping criteria, to train the best model. After the training of the best model was completed, loss curves of training and validation sets were plotted, for comparison with that of the baseline model, as detailed in chapter 4. This was followed by the prediction of PM2.5 values by the best model for the test set and then the evaluation of RMSE score for the test set. After all the evaluations as will be described in the next section, the best model was saved using the Keras model saving function, which saves almost everything about the model including its architecture, optimization, loss function and even weights. This enabled transfer learning as described in the next section. The models were saved in google drive of the researcher.

3.9 Evaluation

From the chapter 2, it can be seen that the most widely used evaluation metric was Root Mean Square Error (RMSE), followed by Mean Absolute Error (MAE), and then by R2 which is also known as the Coefficient of determination. These are elaborated in the glossary attached to the report. So, it was decided to use these three metrics for evaluation of the best models which were done through sklearn python package. In addition, just as Krishan *et al.* (2019) did, the Nash-Sutcliffe Efficiency index (NSE) was also used as a metric for the evaluation of the best models. The evaluation of NSE was done through permetrics python package, so to check the consistency of the results, the R2 was also calculated using permetrics and checked that it matches with the value obtained from sklean. Also, to check the model performance in computational efficiency terms, the training time for the best models was also used as an evaluation metric.

The RMSE and MAE are error-measuring metrics, so the lower their value, the better the model performance. R2 and NSE are accuracy-measuring metrics, so the higher their value, the better the model performance. On the other hand, the lesser the training time of the model, the more computationally efficient it is.

After the training and testing of the best model on Delhi_air_met_df, all the above-mentioned five metrics were evaluated for test data and then the model was saved as described in the previous section. Then the data frames for other cities viz., Bengaluru_air_met_df, Hyderabad_air_met_df, Jaipur_air_met_df and Mumbai_air_met_df were loaded.

Then the saved model was loaded from the google drive of the researcher, using the Keras model loading function. After that, a researcher-defined function was created that would take any of the four above-mentioned data frames, timesteps which were 24 in all cases (because past 24 hours data was

used for 1 hour ahead of future prediction), model and city name as an input, and perform all the actions except for hyperparameter tuning, that were done on Delhi_air_met_df in the previous sections. The function would start from dropping unnecessary columns in the data frame, then splitting data into training and test sets, followed by scaling the data, thereafter time series data preparation, then fitting, training and testing the model and finally produce evaluation metrics of the model along with the plot of loss curve. The loss curves indicate the efficiency of training, help identify overfitting and show how well the model converged. The smoother the curve, the better the model.

As per Weiss, Khoshgoftaar and Wang (2016) transfer learning involves training a model with data from one origin and then utilizing the properties of that model to create another model, that would be further trained with data from a different origin, such that the later has properties obtained from the former as well as that obtained from its latest training (Weiss, Khoshgoftaar and Wang, 2016, pp.3-7).

For each of the four data frames viz., Bengaluru_air_met_df, Hyderabad_air_met_df, Jaipur_air_met_df and Mumbai_air_met_df, the same procedure was done to find the answer to the research question of the project, which is explained as follows.

First, the researcher-defined function was used on each data frame with the best model saved and loaded back, with all its architecture, optimizer settings, loss function and weights. This step was indeed transfer learning, the model already had weights and other information from its previous training. So, this step evaluated the performance of the best model in cities with a lower variance of PM2.5 levels using transfer learning.

A proper evaluation can be said to be done only if a similar evaluation of the best models was done without transfer learning. So, fresh models were created for each city but with all the parameters as that of the best models. Then the researcher-defined function was used on each data frame with this freshly created model. So, this step evaluated the performance of the best model in cities with a lower variance of PM2.5 levels without using transfer learning.

3.10 Conclusion

Hence, applying all the methods described in the above sections of this chapter, the project was carried out to train four RNNs viz., LSTM, BiLSTM, GRU and BiGRU, with the best possible parameters, on the data of a city with a higher variance of PM2.5 levels and the performance of the best models on the data of other cities with a lower variance of PM2.5. This was checked for cases of with transfer learning as well as without transfer learning. The results and their critical evaluation, which are done in the following chapters, would provide an answer to the research question.

Chapter 4 Results

4.1 Initial data pre-processing

As mentioned in chapter 3, the first step of data exploration for pre-processing the data was a boxplot of all columns of station_hr_df2, which is shown in Figure 4.1.1.

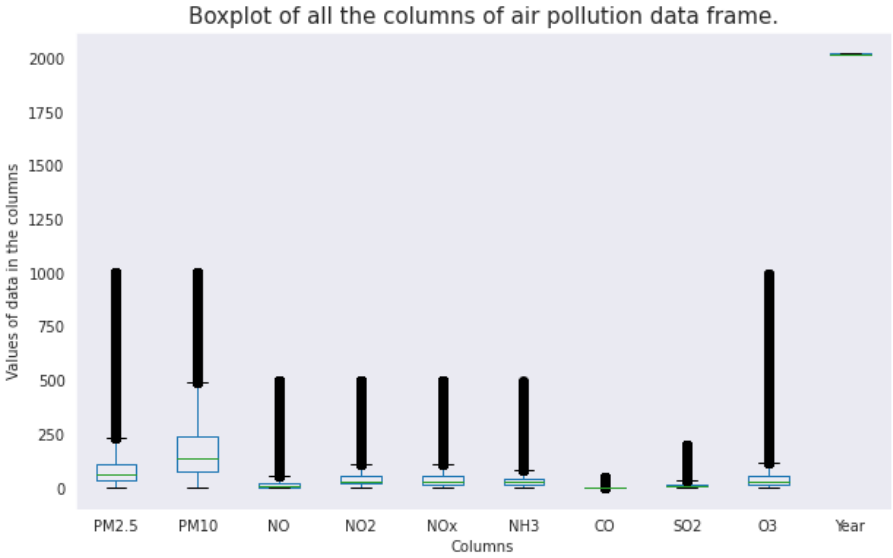


Figure 4.1.1: Boxplot of all columns of station_hr_df2.

As per Figure 4.1.1, the distribution of data in most of the columns of station_hr_df2 is different from one another, although, for the columns NO2 and NOx, there seems to be a similarity in their data distribution. The PM10 column has the widest data distribution as it has the largest gap between the first and third quartiles and also between the minimum and maximum values, followed by PM2.5, whereas the CO column has the thinnest data distribution. All the columns have at least a considerable number of apparent outliers, with O3 having the highest of them.

A comparative analysis was done on distribution plots of all columns of staion_hr_df2 and staion_hr_df2_copy1, i.e., same data, before and after filling missing values by interpolation based on date. As seen in Figure 4.1.2 and Figure 4.1.3, the mean and median of the PM2.5 and PM10 columns seem to be almost unchanged after the imputation of missing values. Similar is the case for other columns as depicted in Figure 4.1.4 to Figure 4.1.10. One thing that is common to all these figures is that all the columns have a log-normal distribution, which is a reason for the use of MinMaxScalar to scale the data, which would normalize the data to have values between 0 and 1. Also, the observations done for various columns from the boxplot in Figure 4.1.1 are consistent with the distribution plots in Figure 4.1.2 to Figure 4.1.10.

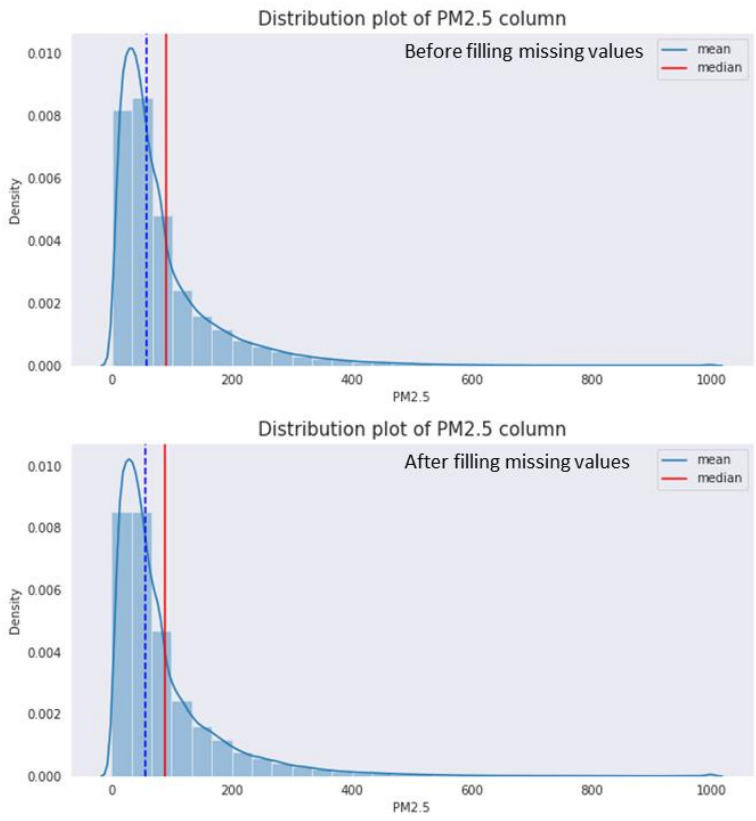


Figure 4.1.2: Distribution plots of PM2.5

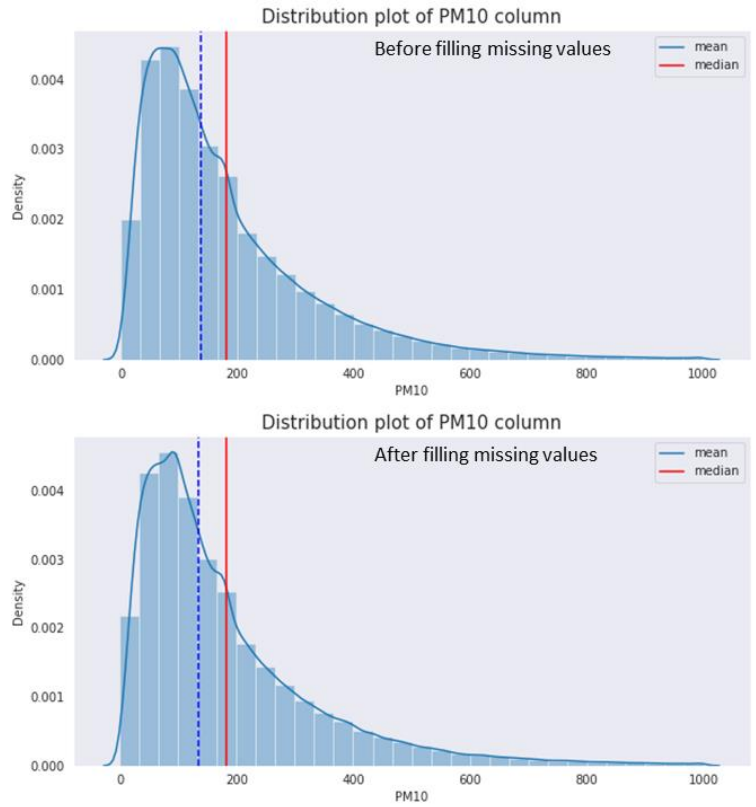


Figure 4.1.3: Distribution plots of PM10

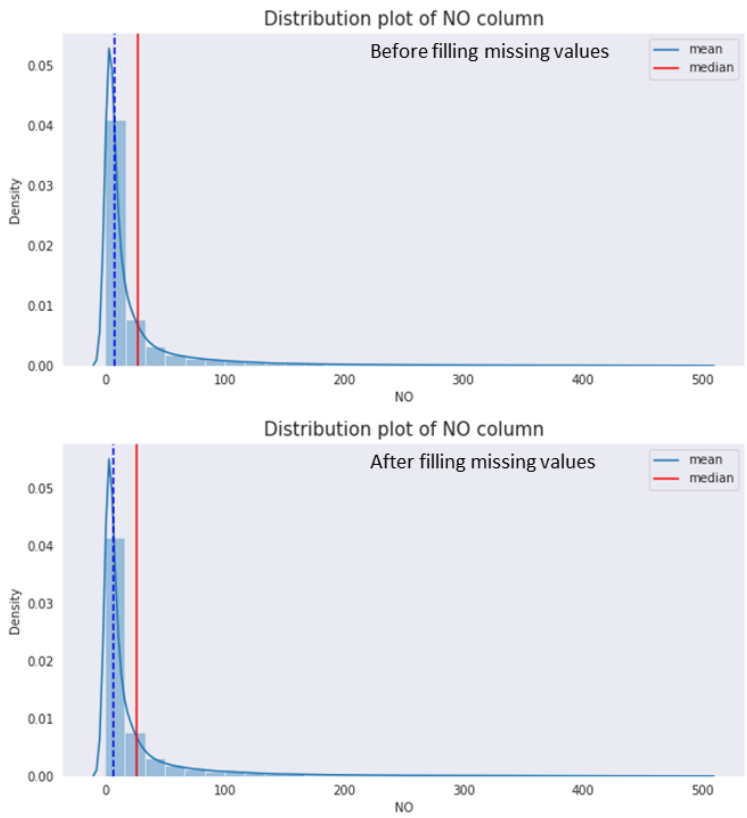


Figure 4.1.4: Distribution plots of NO

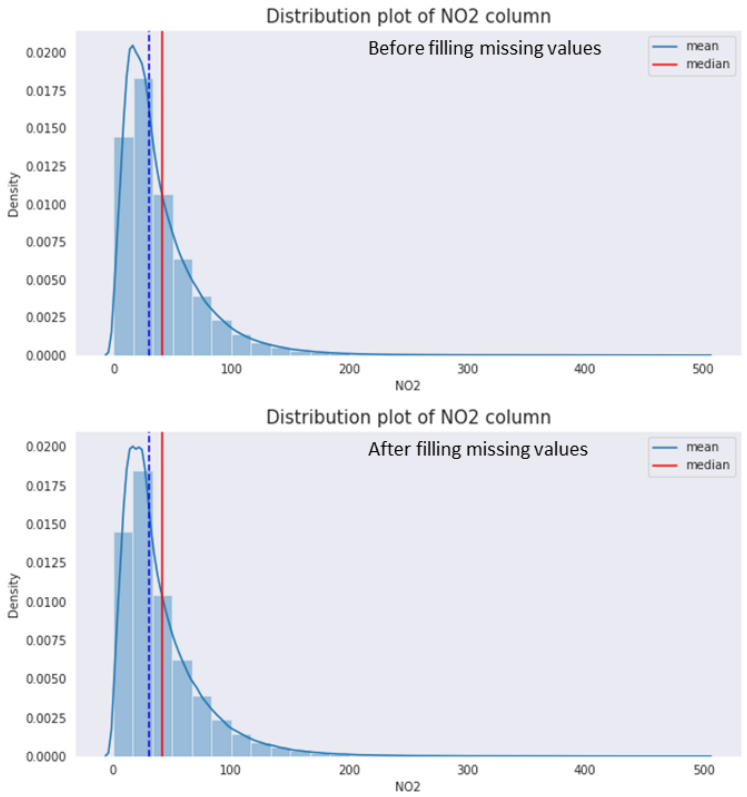


Figure 4.1.5: Distribution of NO2

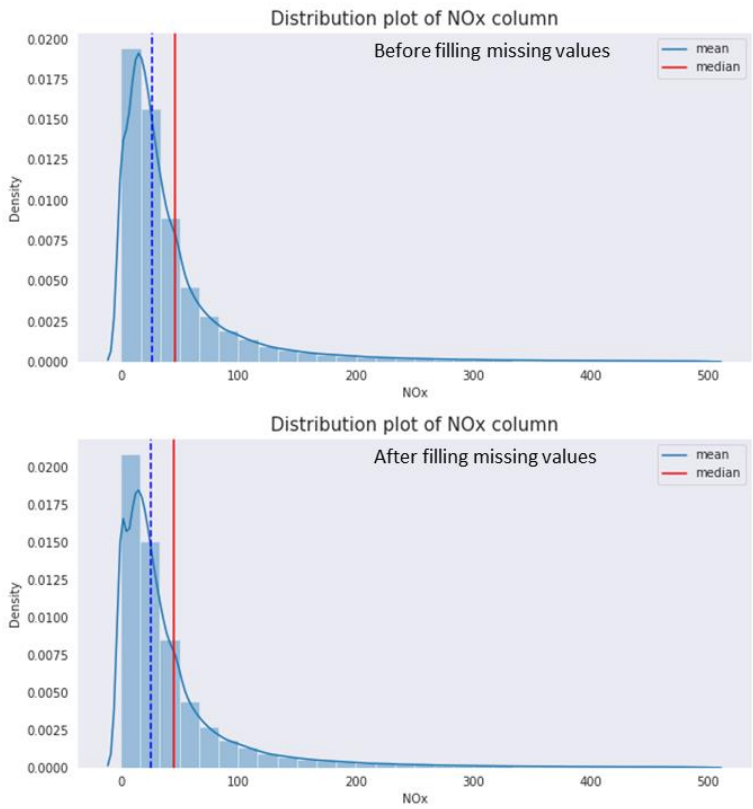


Figure 4.1.6: Distribution plots of NOx

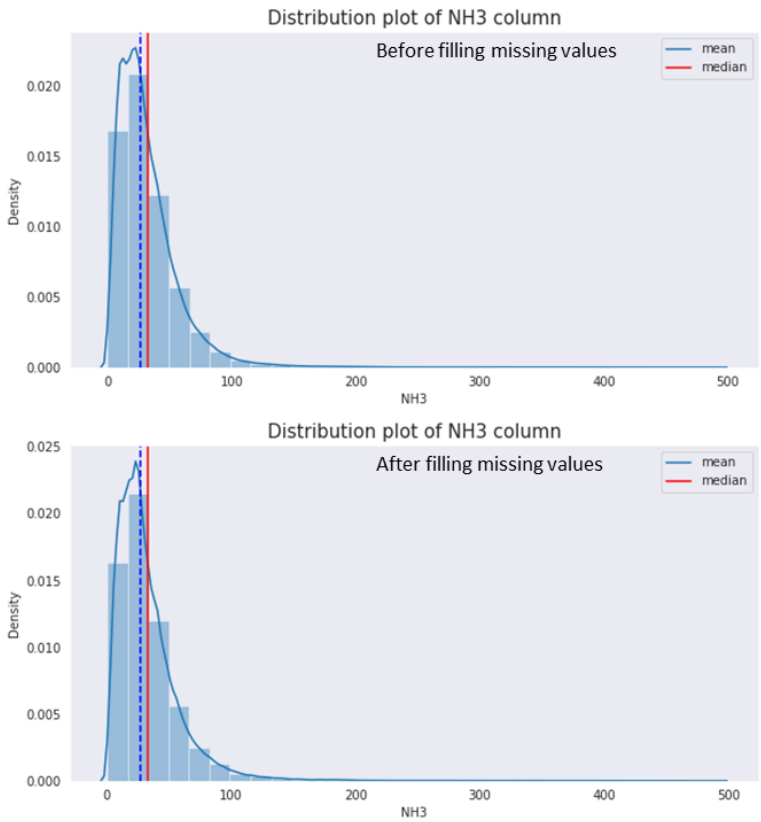


Figure 4.1.7: Distribution plots of NH3

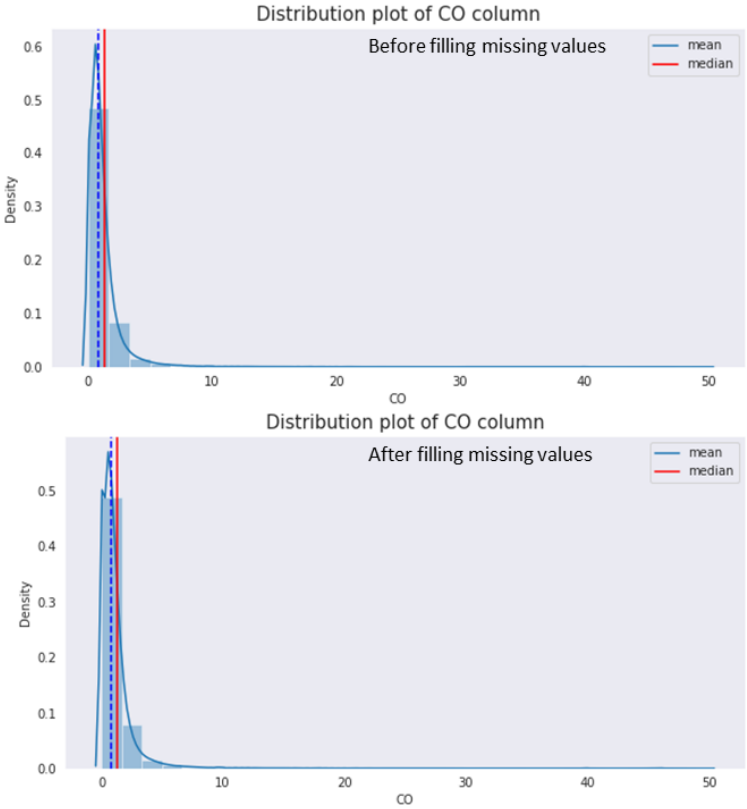


Figure 4.1.8: Distribution plot of CO

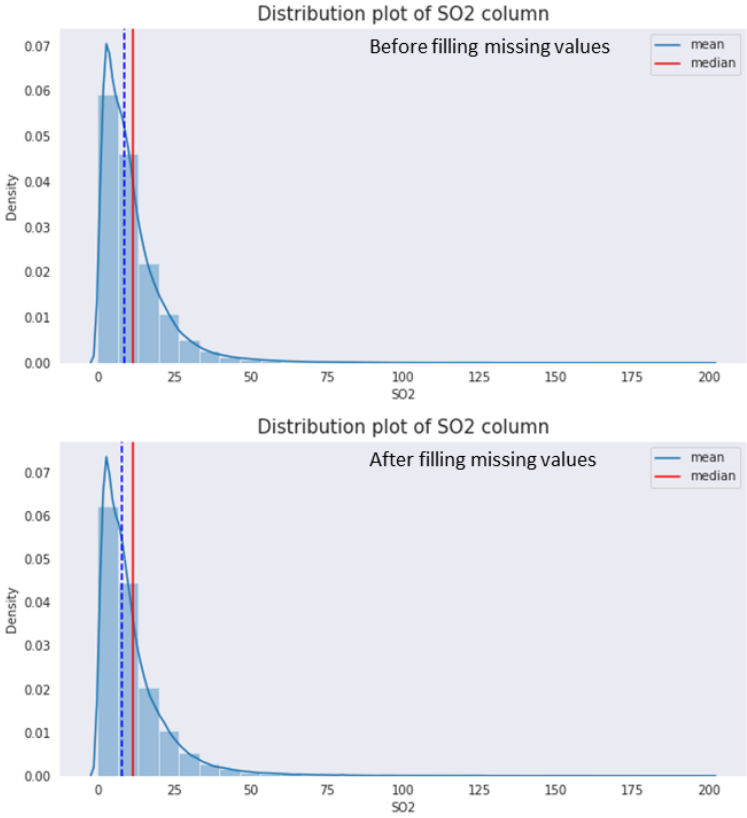


Figure 4.1.9: Distribution plots of SO2

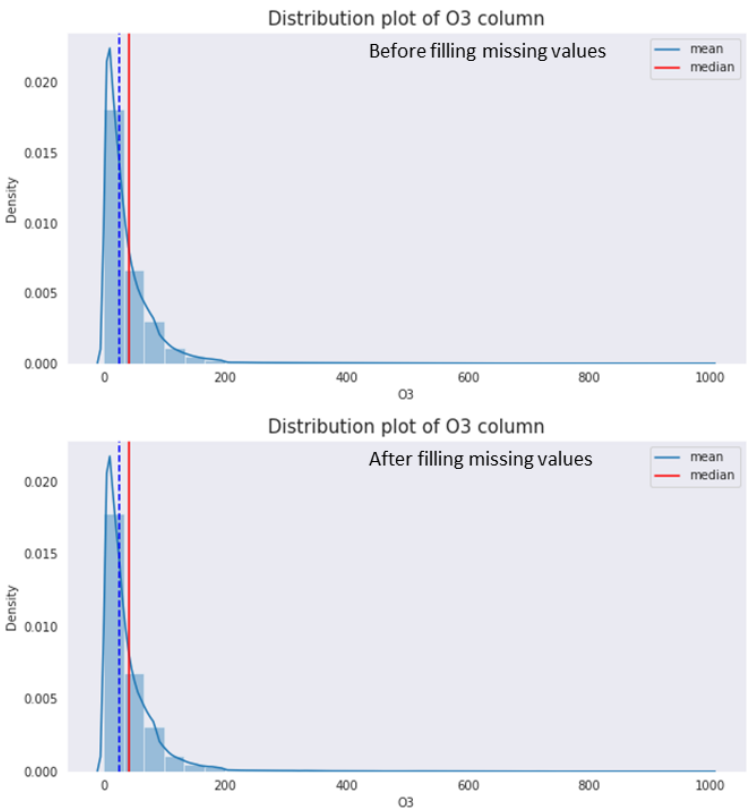


Figure 4.1.10: Distribution plots of O3

Table 4.1.1: Descriptive statistics of station_hf_df2 (with missing values).

Column	***Count***	***Mean***	***Std. dev.***	***Min.***	***25%***	***50%***	***75%***	***Max.***
PM2.5	1081607	90.1598	98.0311	0.0100	31.00	57.9700	110.7500	1000.00
PM10	926814	182.5127	153.2014	0.0100	75.7500	137.00	241.00	1000.00
NO	1138343	26.8198	54.5180	0.0100	3.0600	7.6800	23.3500	500.00
NO2	1155412	41.0044	37.0428	0.0100	17.1300	30.2300	53.3600	499.990
NOx	1216237	46.2679	61.5589	0.00	12.9300	26.7800	52.6100	500.00
NH3	807972	32.3456	26.6458	0.0100	15.200	26.4400	42.0700	494.1100
CO	1187099	1.2793	2.4388	0.00	0.4100	0.8200	1.4200	50.00
SO2	1004384	11.6751	11.9135	0.0100	4.2800	8.5700	15.00	199.9300
O3	1038922	40.9114	55.4994	0.0100	11.0900	25.100	51.8300	997.00

Apart from distribution plots in Figure 4.1.2 to Figure 4.1.10, the descriptive statistics of all columns of staion_hr_df2 and staion_hr_df2_copy1 were also compared, which are indicated in Table 4.1.1 and Table 4.1.2 respectively. Particularly, the changes in mean and standard deviation were observed to check if any statistically big changes have occurred in the data distribution due to the imputation of missing values by interpolation.

Table 4.1.2: Descriptive statistics of station_hf_df2_cop1 (missing values imputed).

Column	***Count***	***Mean***	***Std. dev.***	***Min.***	***25%***	***50%***	***75%***	***Max.***
PM2.5	1502210	89.1360	99.7711	0.0100	30.00	56.8500	109.5700	1000.00
PM10	1502210	182.5335	157.4969	0.0100	74.6900	134.2500	239.50	1000.00
NO	1502210	26.8825	55.6773	0.0100	3.00	7.4500	22.600	500.00
NO2	1502210	41.1563	37.4923	0.0100	17.0700	30.00	53.5700	499.990
NOx	1502210	45.5356	62.8085	0.00	11.5500	25.600	51.6500	500.00
NH3	1502210	32.8581	27.8416	0.0100	15.600	26.4400	42.100	494.1100
CO	1502210	1.2723	2.6312	0.00	0.3600	0.7900	1.400	50.00
SO2	1502210	11.6335	12.4476	0.0100	4.0300	8.1800	14.8300	199.9300
O3	1502210	42.4052	57.8522	0.0100	11.8500	25.9300	53.3500	997.00

Comparing Table 4.1.1 and Table 4.1.2, statistically minor changes have occurred only in the PM2.5, PM10, NOx and O3 columns. For PM2.5, there was a tiny bit of decrease in mean value and a tiny bit of increase in standard deviation, whereas for PM10, the mean remained almost the same but there was an increase in the standard deviation. Similarly, for NOx, there was a tiny bit of decrease in mean value and a tiny bit of increase in standard deviation, whereas, for O3, there was an increase in both mean and standard deviation.

So, from the distribution plots and the descriptive statistics, it was clear that there were close to negligible changes in the distribution of air pollution data due to the imputation of missing values by interpolation based on timestamps.

4.2 Exploratory data analysis of air pollution data

As mentioned in chapter 3, firstly, histogram plots were made to see the variance of PM2.5 levels for different cities, from different points of view.

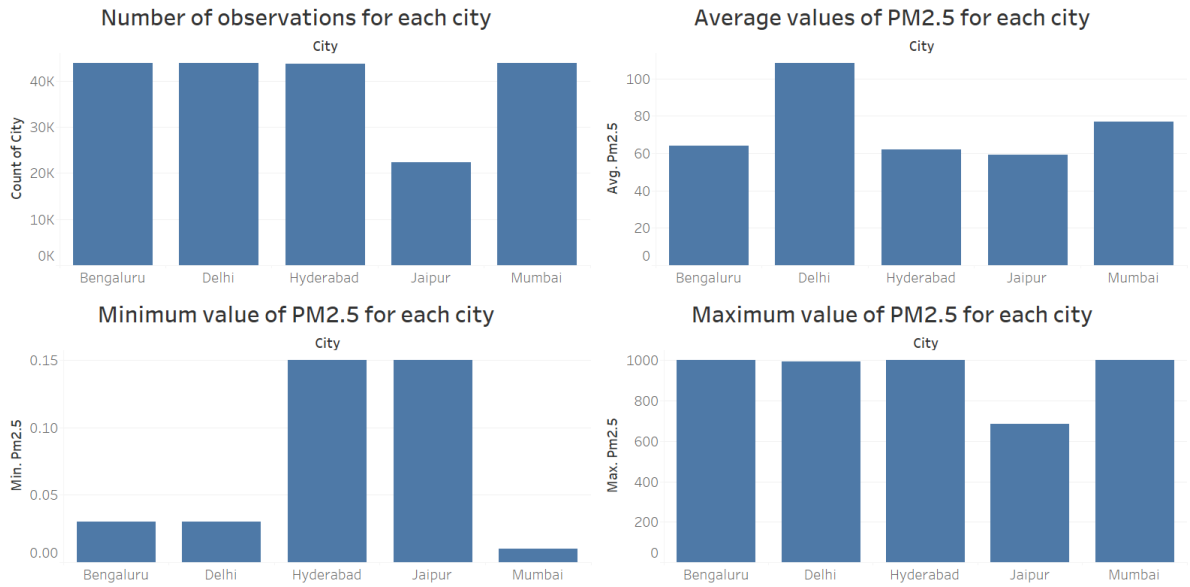


Figure 4.2.1: Histogram plots of PM2.5.

The top left plot of Figure 4.2.1 indicates that the number of observations for Jaipur was almost half of that of the other cities. This is in line with the observations made in Table 3.4.2 of chapter 3. Hence, Jaipur would be significant to see the impact of transfer learning.

The top right plot of Figure 4.2.1 shows that the average value of PM2.5 is the highest for Delhi city, which implies that the mean value of PM2.5 for Delhi city is higher than that of other cities.

The bottom left plot of Figure 4.2.1 reflects that the smallest values of PM2.5 were observed in Mumbai city, followed by Delhi and Bengaluru. On the other hand, the bottom right plot of Figure 4.2.1 reflects that the largest PM2.5 values were almost the same for all cities except for Jaipur.

Collectively looking at all these observations from Figure 4.2.1, it is clear that the variance of PM2.5 values is higher in Delhi city as compared to the other cities. So, it was decided to train, tune and test the models for Delhi city first and then train and test the same models for other cities with transfer learning.

Next, a line plot of per month average PM2.5 levels was plotted which is shown in Figure 4.2.2.

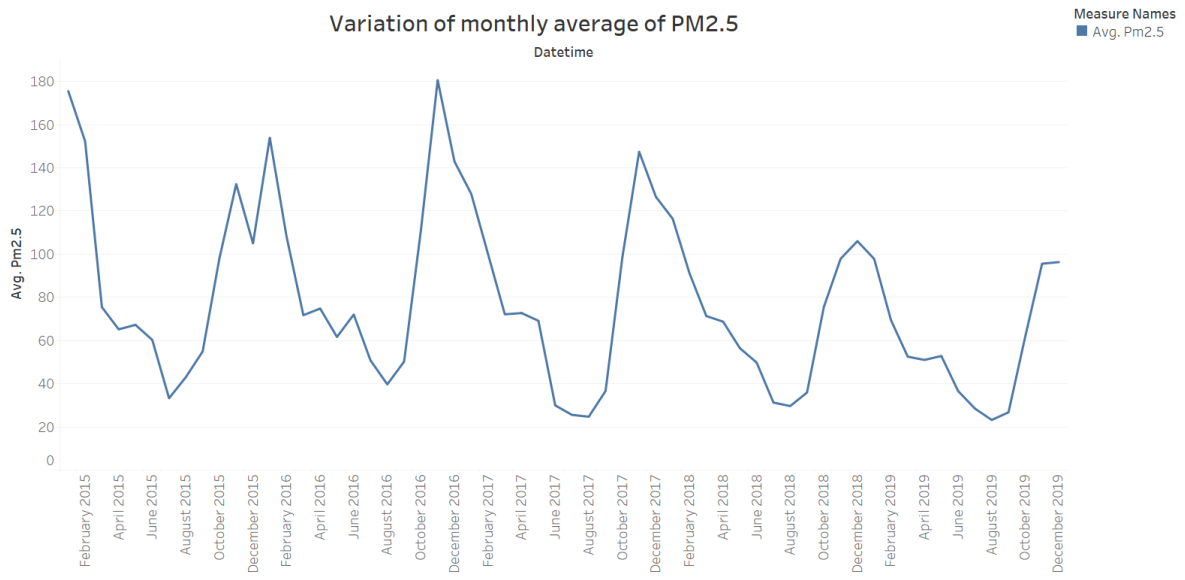


Figure 4.2.2: Line plot of monthly average values of PM2.5

It can be seen from Figure 4.2.2 that there exist some periodicity and seasonality in the levels of PM2.5. The surge to peak levels of PM2.5 was noticed through the winter months of October to December every year, whereas the minimum levels were observed during the rainy months of June to August. From a seasonality point of view, these trends might be attributed to meteorological factors like temperature and rainfall to aid the RNNs for prediction.

Then, a similar plot of per month average PM2.5 levels along with those of other pollutants was plotted, which is displayed in Figure 4.2.3 and Figure 4.2.4.

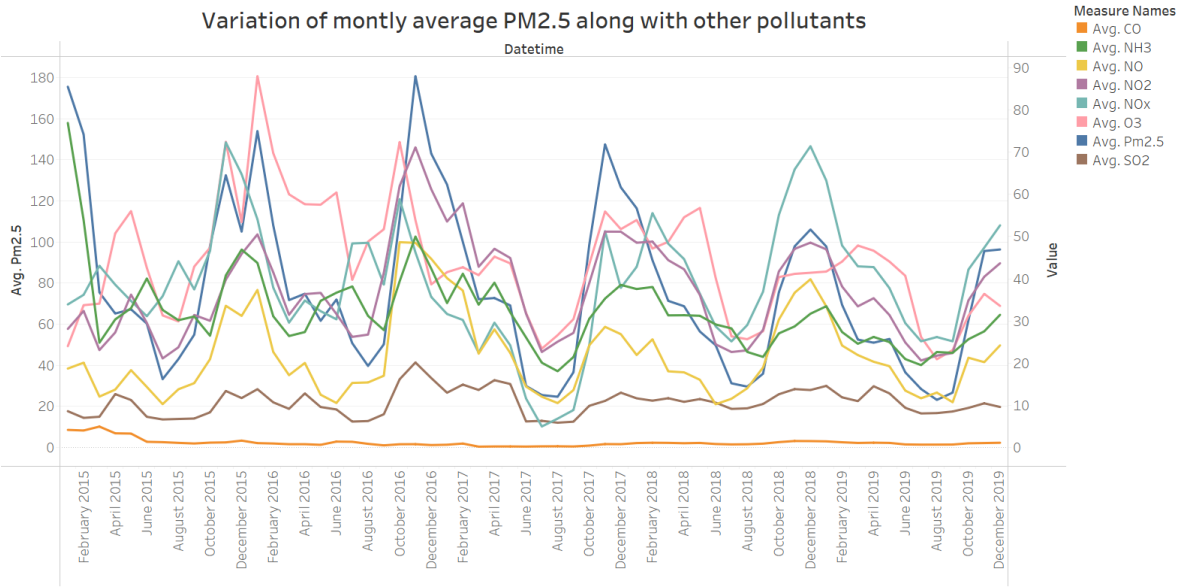


Figure 4.2.3: Line plot of monthly average values of PM2.5 with other pollutants.

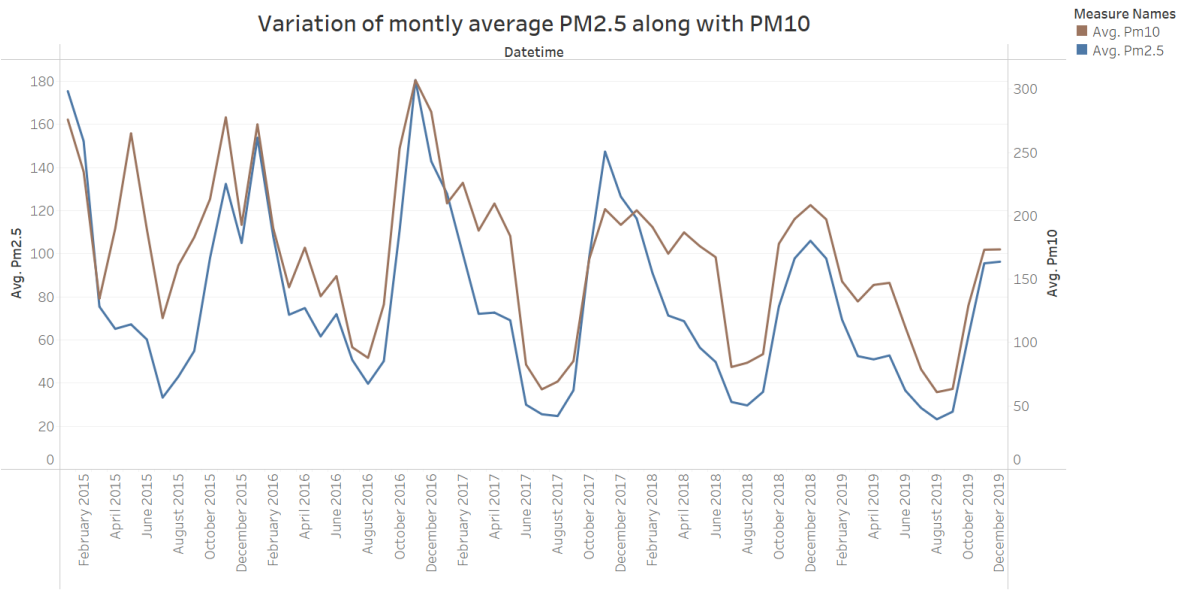


Figure 4.2.4: Line plot of monthly average values of PM2.5 with PM10

From Figure 4.2.3 it can be observed that all the nitrogen-based pollutants viz., NO, NO2 and NOx follow the pattern of emission of PM2.5. NO almost replicated the emission trend of PM2.5 at every change in their concentrations, which may imply a high correlation between them and also, they might influence each other. NO2 and NOx appear to have a similar correlation with PM2.5. The pattern of O3 seemed to be similar to that of PM2.5 during the years 2015 and 2016, after which it started to be very different from the pattern of PM2.5. CO is the only pollutant that seems to have no relation to the emission trend of PM2.5 and was almost stable throughout the study period of five years.

The trend shown in Figure 4.2.4 indicates that PM2.5 has the closest relation to PM10. It is obvious as both are particulate matter but of different sizes. One thing to note is that except for the period between October 2017 to February 2018, levels of PM10 were always higher than that of PM2.5, which may mean that some of the PM2.5 might have been carried by PM10.

Hence, from the line plots, it can be said that the emission of other pollutants except for CO has a considerable relation with that of PM2.5 and they all are necessary for predicting PM2.5 levels.

The final step of exploratory data analysis of pollution data was the preparation of scatter plots of PM2.5 against other pollutants, which are depicted in Figure 4.2.5 and Figure 4.2.6.

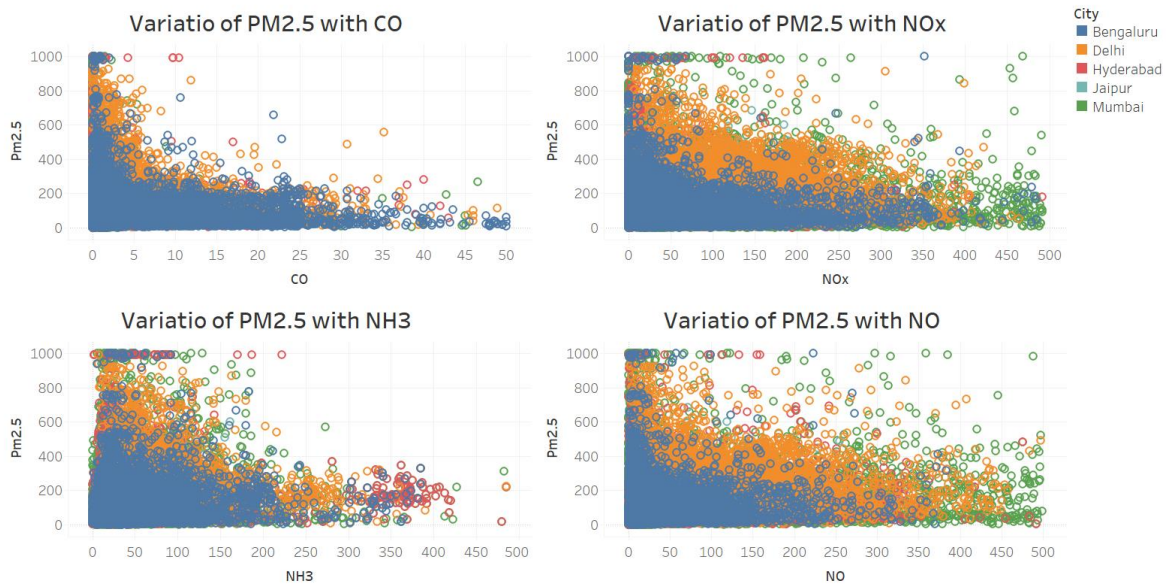


Figure 4.2.5: Scatter plots of PM2.5 vs other pollutants.

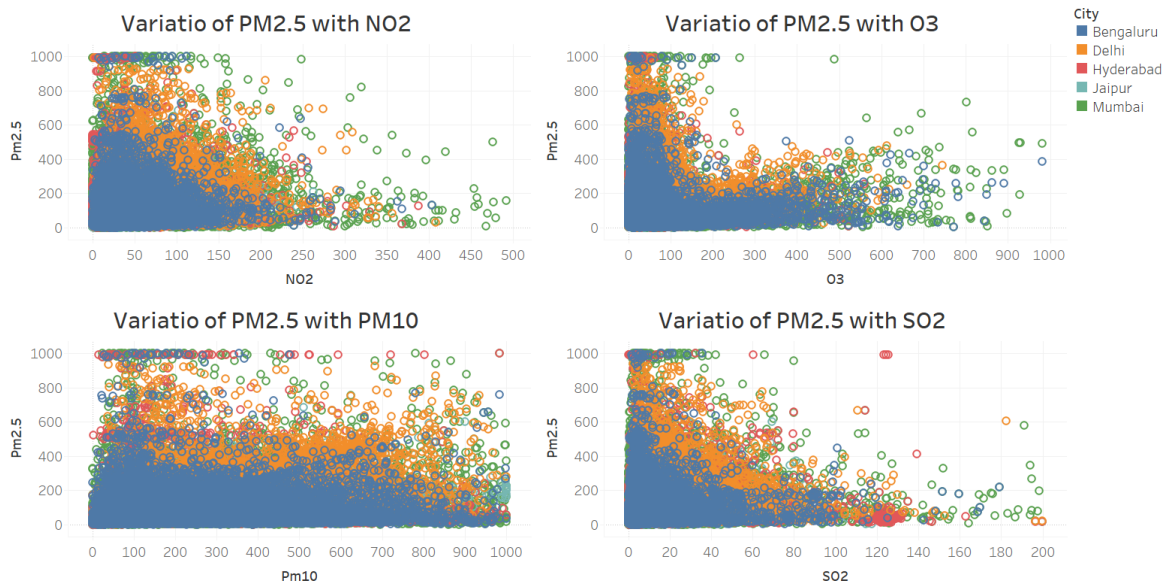


Figure 4.2.6: Scatter plots of PM2.5 vs other pollutants.

Figure 4.2.5 and Figure 4.2.6 show that there is no formation of a cluster in the air pollution data. The different colours of dots in the scatter plot represent data points of different cities. So, there seems to be no spatial segregation or grouping in the data, i.e., it cannot be specified that a particular city has air pollution emission levels within a particular range. Therefore, no strong reason is visible to consider the spatial aspect of pollutants for further analysis and modelling.

Also, from all the scatter plots, almost all the pollutants have a negative correlation with PM2.5, especially for NO, NO2 and NOx there appears to be a strong negative correlation.

Hence, from the results of exploratory data analysis, it was determined that Delhi city has a higher variance in PM2.5 levels than other cities, which has a considerable correlation with almost all the pollutants. So, it was decided by the researcher that all the further pre-processing will also be done based on the necessary air pollution and meteorological data for Delhi city. Also, the spatial aspect doesn't seem to have any influence on the emission levels of pollutants.

4.3 Further data pre-processing

The first result expected in further data pre-processing was the correlation matrix of Delhi_air_met_df, which is indicated in Figure 4.3.1 and Figure 4.3.2.

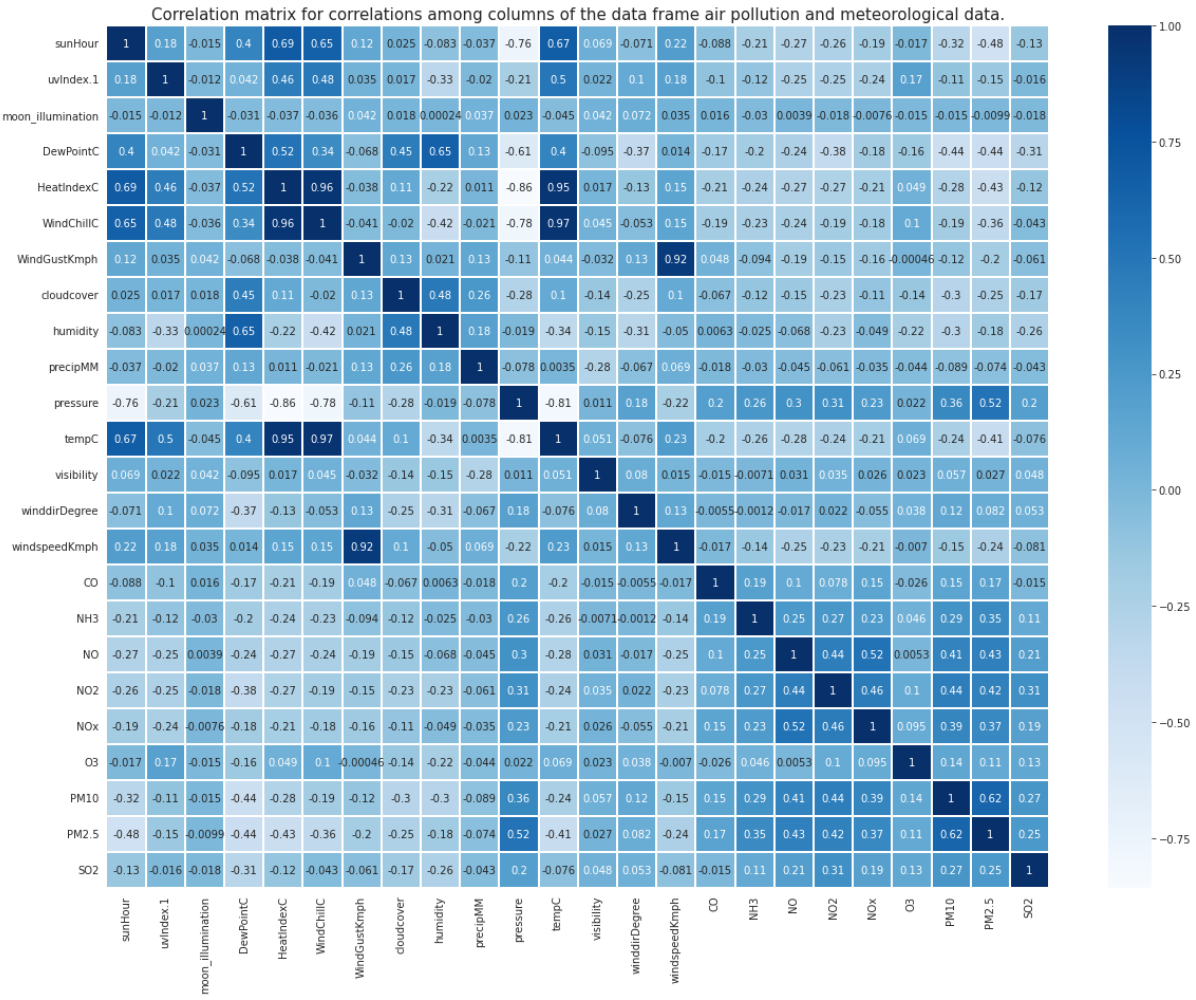


Figure 4.3.1: Correlation matrix of Delhi_air_met_df (initial).

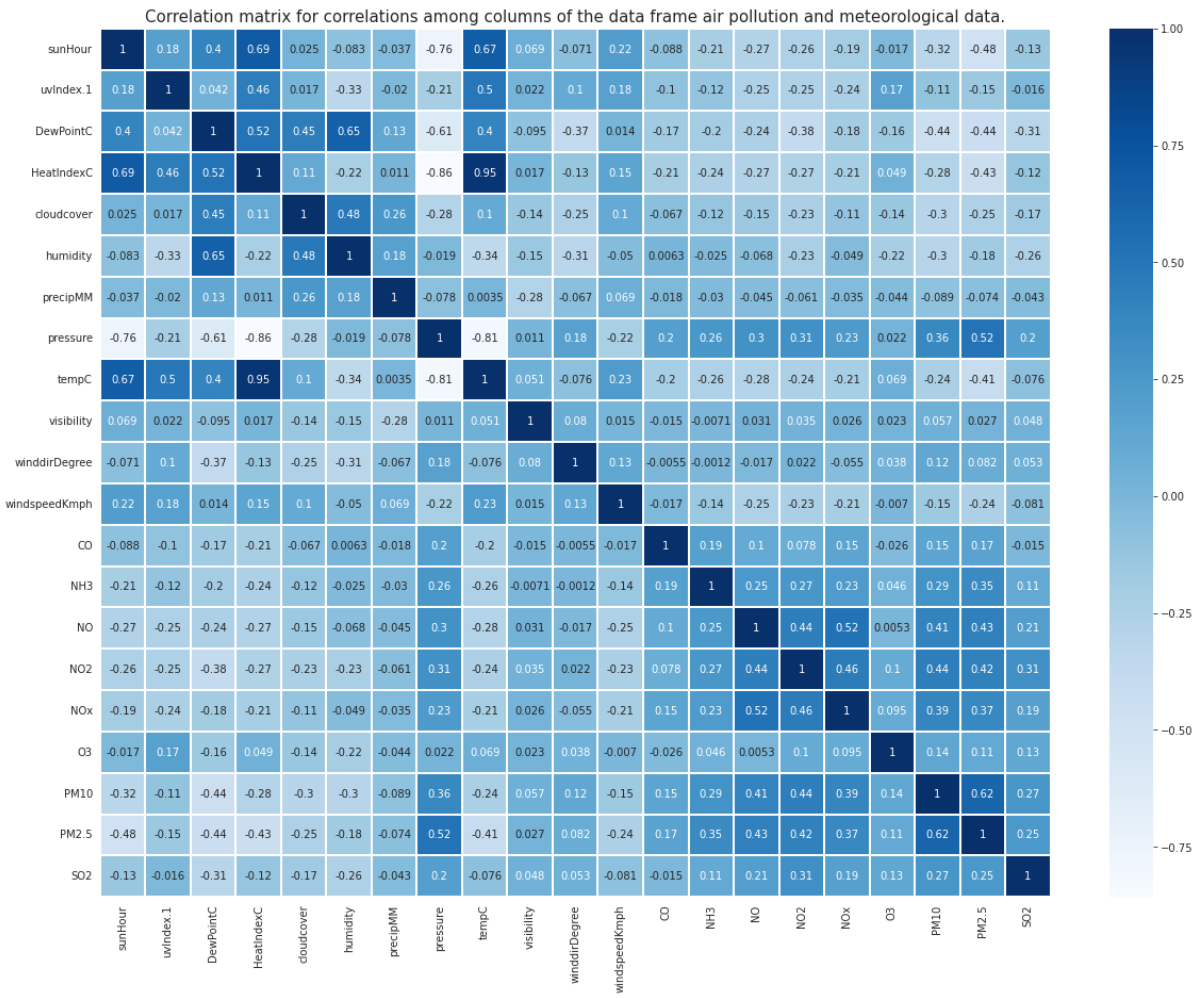


Figure 4.3.2: Correlation matrix of Delhi_air_met_df (final).

In Figure 4.3.1, it can be found that the column WindChillC had very high correlations of 0.97 and 0.96 with tempC and HeatIndexC respectively. Similarly, HeatIndexC and tempC had a correlation of 0.95. Apart from these, WindGustKmph and WindSpeedKmph also had a high correlation of 0.92. The presence of all these columns would be redundant for prediction and feature selection must be done to remove some of them.

The column tempC is a very important factor, so there was no question of dropping it. Out of WindChillC and HeatIndexC, the latter had a better correlation with PM2.5 and would be more important for models to predict PM2.5. Similarly, out of WindGustKmph and WindSpeedKmph, the latter had a better correlation with PM2.5 and would be more important for models to predict PM2.5. Out of all the columns, the moon_illumination column had an almost negligible correlation of -0.0099 with PM2.5 and it won't be of any advantage having this column for models to predict PM2.5. Based on these reasons, WindChillC, WindGustKmph and moon_illumination columns were dropped from Delhi_air_met_df.

After dropping all those columns, the correlation matrix is represented by Figure 4.3.2, in which a high correlation still exists between tempC and HeatIndexC, but both have a good correlation with PM2.5 and hence were not dropped.

Hence, the results of the correlation matrix helped in feature selection and enabled to drop four columns from the data that would not help the models greatly to predict PM2.5.

Next in the line of data pre-processing was to try to identify the outliers in the data, for which the descriptive statistics of all the columns as described in Table 4.3.1 were checked.

Table 4.3.1: Descriptive statistics of Delhi_air_met_df.

Column	**Count**	**Mean**	**Std**	**Min**	**25%**	**50%**	**75%**	**Max**
sunHour	43823	11.135	2.177	3.800	8.700	11.600	13.200	14.000
uvIndex.1	43823	4.467	3.416	1.000	1.000	5.000	8.000	11.000
DewPointC	43823	11.549	8.156	-22.000	5.000	11.000	19.000	29.000
HeatIndexC	43823	31.349	9.147	8.000	24.000	32.000	39.000	61.000
cloudcover	43823	15.690	21.647	0.000	0.000	5.000	24.000	100.000
humidity	43823	37.078	18.417	2.000	23.000	35.000	50.000	99.000
precipMM	43823	0.037	0.324	0.000	0.000	0.000	0.000	17.600
pressure	43823	1007.21	7.752	987.00	1000.00	1007.00	1014.00	1025.00
tempC	43823	27.876	8.447	0.000	21.000	29.000	34.000	51.000
visibility	43823	10.015	1.043	0.000	10.000	10.000	10.000	20.000
winddirDegree	43823	213.01	99.689	0.000	117.000	252.000	300.000	360.000
windspeedKmph	43823	9.202	4.562	0.000	6.000	9.000	12.000	44.000
CO	43823	1.874	2.869	0.000	0.664	1.047	1.756	48.880
NH3	43823	38.178	23.719	0.570	25.159	33.106	44.675	485.520
NO	43823	33.285	45.312	0.180	9.391	16.209	36.619	497.400
NO2	43823	47.011	27.610	0.850	27.810	40.397	59.481	408.021
NOx	43823	54.631	47.236	0.000	25.348	40.003	69.432	463.878
O3	43823	48.801	40.744	0.060	24.108	38.241	62.475	744.503
PM10	43823	209.32	142.962	0.450	106.331	175.460	275.879	999.750
SO2	43823	13.142	10.070	0.020	7.356	10.668	15.900	199.930
PM2.5	43823	108.235	94.244	0.030	46.220	78.723	141.827	995.000

From Table 4.3.1 it can be seen that cloudcover, precipMM, windspeedKmph, CO, NH3, NO, NO2, NOx, O3, PM10, SO2 and PM2.5 are the columns for which the maximum values are more than double the mean and median values of the columns. Also, the difference between the 75 percentile values and

the maximum values varies differently for different columns. This implies statistically that there could be outliers present in these columns.

As a further step of outlier detection, the median values, the 99 percentile values and thrice the standard deviation, described in Table 4.3.2 were checked.

Table 4.3.2: Outlier detection parameters.

Column	***Median Value***	***99 percentile values***	***Thrice Std. deviation***
sunHour	11.600	14.000	6.531
uvIndex.1	5.000	10.000	10.248
DewPointC	11.000	25.000	24.469
HeatIndexC	32.000	50.000	27.442
cloudcover	5.000	88.000	64.941
humidity	35.000	84.000	55.252
precipMM	0.000	0.900	0.972
pressure	1007.000	1021.000	23.257
tempC	29.000	44.000	25.342
visibility	10.000	14.000	3.129
winddirDegree	252.000	351.000	299.067
windspeedKmph	9.000	23.000	13.685
CO	1.047	14.194	8.606
NH3	33.106	125.529	71.158
NO	16.209	226.831	135.936
NO2	40.397	139.865	82.831
NOx	40.003	230.762	141.709
O3	38.241	215.914	122.233
PM10	175.460	691.034	428.885
SO2	10.668	52.416	30.210
PM2.5	78.723	450.504	282.731

From Table 4.3.2 it can be seen that except for uvIndex.1 and precipMM columns, the rest of the columns don't have 99 percentile of their data within thrice of standard deviation. Statistically, it is an indication of the presence of outliers in those columns. Collectively looking at the observations from Table 4.3.1 and Table 4.3.2, it is difficult the set a common criterion for the removal of outliers as the data distribution between 75 percentile and 99 percentile values is different for all the columns. So, as mentioned in chapter 3, there was no outlier removal for these reasons.

4.4 Exploratory data analysis of air pollution and meteorological data

Before going for modelling, some exploratory data analysis was done to identify meaningful patterns and relations between PM2.5 and meteorological data for all the cities. This was done by preparing line plots of per month average PM2.5 levels and all the meteorological factors except for WinddirDegree, for which a scatter plot was prepared. This was done for the data of all the five cities considered in the current project.

The line plots and scatter plots for Delhi city, the data of which will be used for creating models, are as shown in Figure 4.4.1, Figure 4.4.2 and Figure 4.4.3.

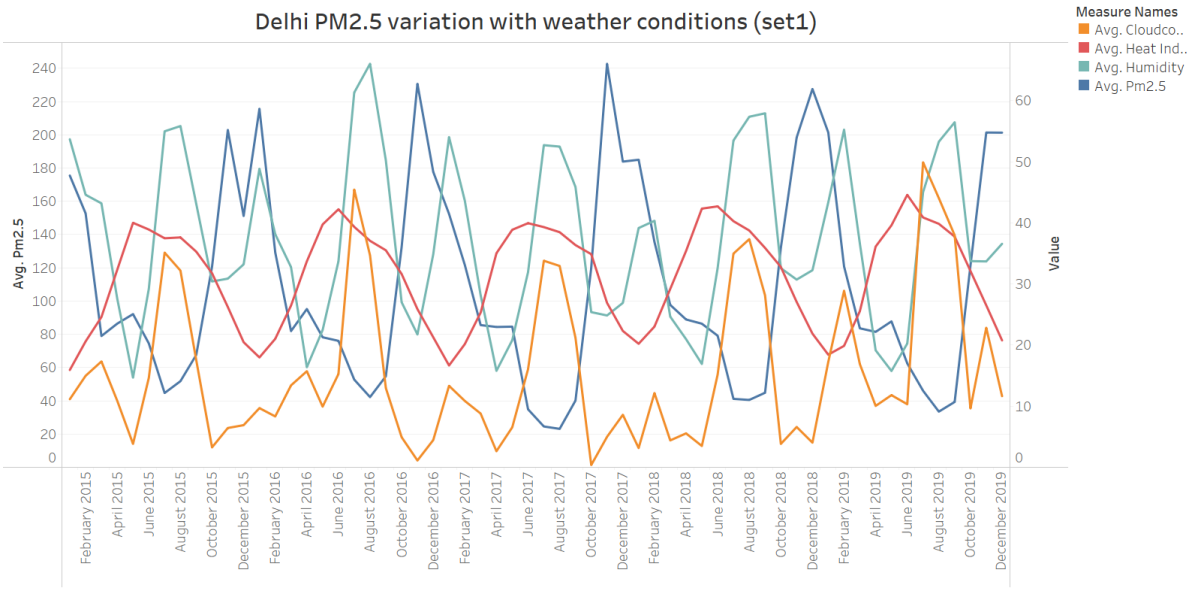


Figure 4.4.1: PM2.5 variation with Meteorological factors (Delhi).

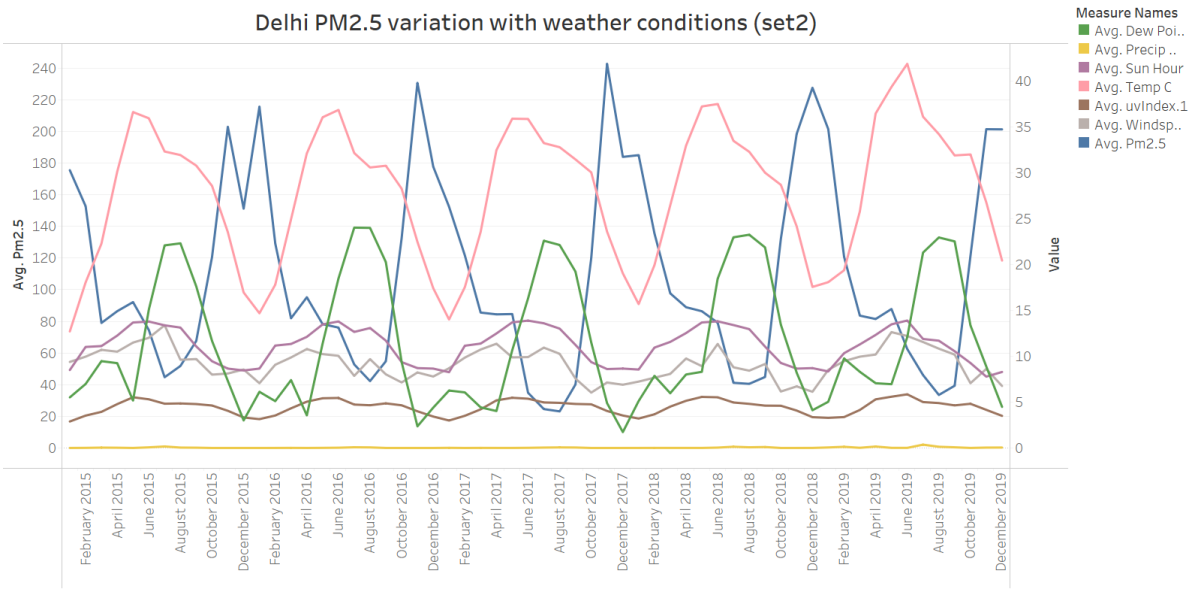


Figure 4.4.2: PM2.5 variation with Meteorological factors (Delhi).

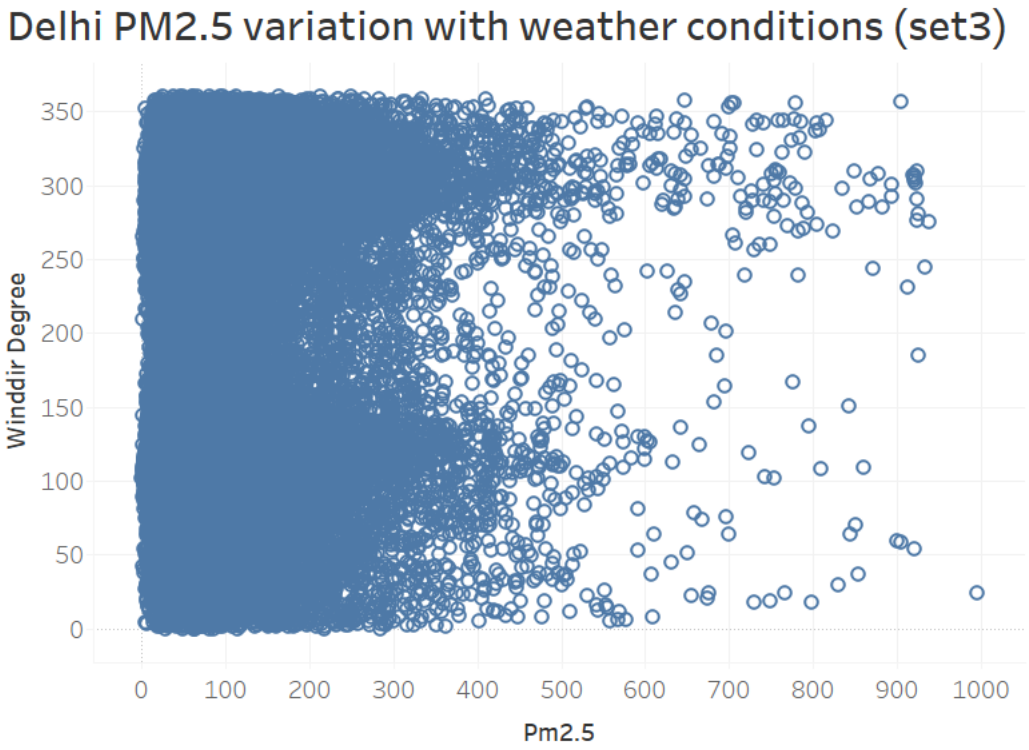


Figure 4.4.3:PM2.5 variation with WinddirDegree (Delhi).

As can be seen in Figure 4.4.1, Figure 4.4.2 and Figure 4.4.3, none of the meteorological factors have any relation to the emission pattern of PM2.5. Although the correlation matrix from an earlier section indicated the existence of correlations between PM2.5 and all these meteorological factors, all the plots in Figure 4.4.1 to Figure 4.4.3 show that none of their patterns has any similarity to that of PM2.5.

A similar pattern was shown for the rest of the cities as well and there were no additional insights from them, which is why they haven't been produced here in this chapter, but all those plots are presented in appendix A attached to the report.

4.5 Training, tuning, testing and creation of best models

All the training, tuning, testing and creation were done on the data of Delhi city as it was concluded in earlier sections of this chapter that it is the city with a higher variance of PM2.5 level compared to other cities. This choice is in line to answer the research question of the current project.

4.5.1 LSTM

The results of LSTM models' performance for test data, starting from the baseline model to the best model are mentioned in Table 4.5.1. Note that the output layer was a dense layer of one neuron for all models which is why it is not mentioned anywhere. All the layers in Table 4.5.1 are of LSTM.

Table 4.5.1: Results of LSTM model in progression from baseline to best model.

Model	***Tunning method***	***Model parameters***	***Test RMSE***
Model0 (baseline)	-	1 layer, 64 neurons, learning rate = 0.01	20.442
Model11	Hyperband	1 layer, 50 neurons, learning rate = 0.01	20.641
Model12	Bayesian optimization	1 layer, 100 neurons, learning rate = 0.001	22.413
Model13	Hyperband	1 layer, 70 neurons, learning rate = 0.001	20.156
Model14	Bayesian optimization	1 layer, 50 neurons, learning rate = 0.001	23.787
Model15	Hyperband	1 layer, 65 neurons, learning rate = 0.01	23.791
Model16	Bayesian optimization	1 layer, 75 neurons, learning rate = 0.0001	23.587
Model21	Hyperband	Layer 1, 50 neurons, Layer 2, 50 neurons, Layer 3, 40 neurons, Layer 4, 40 neurons, learning rate = 0.001	23.034
Model22	Bayesian optimization	Layer 1, 50 neurons, Layer 2, 60 neurons, learning rate = 0.01	20.389
Model23	Hyperband	Layer 1, 45 neurons, Layer 2, 45 neurons, learning rate = 0.01	20.508
Model24	Bayesian optimization	Layer 1, 60 neurons, Layer 2, 55 neurons, learning rate = 0.0001	22.250
Model17	-	1 layer, 70 neurons, learning rate = 0.001, activation = ‘sigmoid’	27.540
Model18	-	1 layer, 70 neurons, learning rate = 0.001, activation = ‘relu’	23.456
Model31	Hyperband	1 layer, 70 neurons, learning rate = 0.001, Dropout = 0.4	23.550
Model32	Bayesian optimization	1 layer, 70 neurons, learning rate = 0.001, Dropout = 0.7	26.097

Model41	Hyperband	1 layer, 70 neurons, learning rate = 0.001, Dropout = 0.4, weight decay = 0.001	27.610
Model42	Bayesian optimization	1 layer, 70 neurons, learning rate = 0.001, Dropout = 0.4, weight decay = 0.0001	24.201
Model51	-	1 layer, 70 neurons, learning rate = 0.001, Dropout = 0.4, weight decay = 0.0001, Batch size = 32	28.325
Model52	-	1 layer, 70 neurons, learning rate = 0.001, Dropout = 0.4, weight decay = 0.0001, Batch size = 128	23.458
Model53	-	1 layer, 70 neurons, learning rate = 0.001, Dropout = 0.4, weight decay = 0.0001, Batch size = 192	24.338

After training and testing the baseline model (Model0), hyperparameter tuning was done using Hyperband as well as Bayesian-Optimization methods to search for an optimal number of neurons and an optimal value of learning rate for a single layer LSTM. For this, all models from Model11 to Model16 were trained and tested. Then, a search was done for an optimal number of layers, an optimal number of neurons and an optimal value of learning rate for a multilayer LSTM, from Model21 to Model 24. Based on test RMSE scores of all models from Model11 to Model24, Model13 with a single layer of 70 neurons and a learning rate of 0.001 gave the best performance. So, the parameters of this model were considered for the best model.

Next, a comparison of the performance of Model17 and Model18 with Model13 showed that the default activation of tanh was the optimal activation function. From the results of Model31 and Model32, it was concluded that a dropout value of 0.4 was optimal to regularize the model. Similarly, from the results of Model41 and Model42, a weight decay value of 0.0001 was optimal to further regularize the model. Finally, the performance of Model51, Model52 and Model53 indicated that training the models with a batch size of 128 gave the best performance. So, all of these parameters were considered for the best model.

With all the optimal parameters obtained above, the best LSTM model was created, trained, tested and evaluated. The details of the best LSTM parameters and performance metrics are described in Table 4.5.2.

Table 4.5.2: Best LSTM parameters and Performance metrics.

Model	***Parameters***	***Evaluation metric***	***Value/ Score***
Best LSTM	Number of layers = 1, number of neurons = 70,	Training time	293. 76 sec
	Dropout = 0.4, weight decay = 0.0001,	RMSE	24.491
	Learning rate = 0.001, epochs = 100,	MAE	15.149
	Batch size = 128, early stopping patience = 10	R2	0.931
		NSE	0.929

Figure 4.5.1 and Figure 4.5.2 show the loss curves of the baseline LSTM model and Best LSTM model respectively.

Figure 4.5.1:Baseline LSTM.

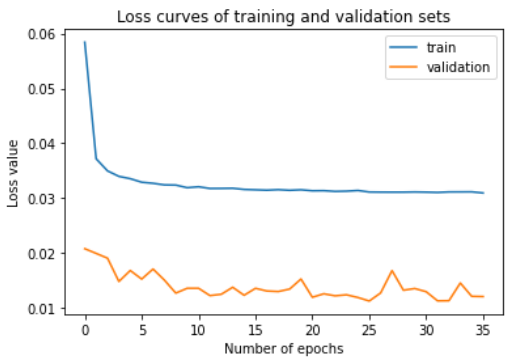


Figure 4.5.2: Best LSTM.

Comparing Figure 4.5.1 and Figure 4.5.2, loss curves of both training and validation sets for the best LSTM model are smoother and more stable than those of the baseline LSTM model, as a result of hyperparameter tuning and addition of regularization parameters.

4.5.2 Bidirectional LSTM or BiLSTM

The results of BiLSTM models performance for test data, starting from the baseline model to the best model are mentioned in Table 4.5.3. Note that the output layer was a dense layer of one neuron for all models which is why it is not mentioned anywhere. All the layers in Table 4.5.3 are of BiLSTM.

Table 4.5.3: Results of BiLSTM model in progression from baseline to best model.

Model	***Tunning method***	***Model parameters***	***Test RMSE***
Model0 (baseline)	-	1 layer, 64 neurons, learning rate = 0.01	19.768
Model11	Hyperband	1 layer, 50 neurons, learning rate = 0.0001	22.746
Model12	Bayesian optimization	1 layer, 150 neurons, learning rate = 0.0001	22.193
Model13	Hyperband	1 layer, 70 neurons, learning rate = 0.001	22.099
Model14	Bayesian optimization	1 layer, 80 neurons, learning rate = 0.01	20.668
Model15	Hyperband	1 layer, 75 neurons, learning rate = 0.01	20.938
Model16	Bayesian optimization	1 layer, 70 neurons, learning rate = 0.01	23.182
Model21	Hyperband	Layer 1, 70 neurons, Layer 2, 60 neurons, learning rate = 0.01	23.928
Model22	Bayesian optimization	Layer 1, 50 neurons, Layer 2, 70 neurons, Layer 3, 40 neurons, learning rate = 0.001	22.960
Model23	Hyperband	Layer 1, 50 neurons, Layer 2, 60 neurons, learning rate = 0.001	21.609
Model24	Bayesian optimization	Layer 1, 50 neurons, Layer 2, 45 neurons, learning rate = 0.01	20.837
Model17	-	1 layer, 64 neurons, learning rate = 0.01, activation = ‘sigmoid’	22.019
Model18	-	1 layer, 64 neurons, learning rate = 0.01, activation = ‘relu’	24.265
Model31	Hyperband	1 layer, 64 neurons, learning rate = 0.01, Dropout = 0.5	24.130
Model32	Bayesian optimization	1 layer, 64 neurons, learning rate = 0.01, Dropout = 0.1	23.471
Model41	Hyperband	1 layer, 64 neurons, learning rate = 0.01,	34.230

		Dropout = 0.1, weight decay = 0.01	
Model42	Bayesian optimization	1 layer, 64 neurons, learning rate = 0.01, Dropout = 0.1, weight decay = 0.0001	22.975
Model51	-	1 layer, 64 neurons, learning rate = 0.01, Dropout = 0.1, weight decay = 0.0001, Batch size = 32	24.892
Model52	-	1 layer, 64 neurons, learning rate = 0.01, Dropout = 0.1, weight decay = 0.0001, Batch size = 128	24.986
Model53	-	1 layer, 64 neurons, learning rate = 0.01, Dropout = 0.1, weight decay = 0.0001, Batch size = 192	20.965

After training and testing the baseline model (Model0), hyperparameter tuning was done using Hyperband as well as Bayesian-Optimization methods to search for an optimal number of neurons and an optimal value of learning rate for a single layer BiLSTM. For this, all models from Model11 to Model16 were trained and tested. Then, a search was done for an optimal number of layers, an optimal number of neurons and an optimal value of learning rate for a multilayer BiLSTM, from Model21 to Model 24. Based on test RMSE scores of all models from Model11 to Model24, Model0 or the baseline model with a single layer of 64 neurons and a learning rate of 0.01 gave the best performance. So, the parameters of this model were considered for the best model.

Next, a comparison of the performance of Model17 and Model18 with Model13 showed that the default activation of tanh was the optimal activation function. From the results of Model31 and Model32, it was concluded that a dropout value of 0.1 was optimal to regularize the model. Similarly, from the results of Model41 and Model42, a weight decay value of 0.0001 was optimal to further regularize the model. Finally, the performance of Model51, Model52 and Model53 indicated that training the models with a batch size of 192 gave the best performance. So, all these parameters were considered for the best model.

With all the optimal parameters obtained above, the best BiLSTM model was created, trained, tested and evaluated. The details of the best BiLSTM parameters and performance metrics are described in Table 4.5.4.

Table 4.5.4: Best BiLSTM parameters and Performance metrics.

Model	***Parameters***	***Evaluation metric***	***Value/ Score***
Best BiLSTM	Number of layers = 1, number of neurons = 64, Dropout = 0.1, weight decay = 0.0001, Learning rate = 0.01, epochs = 100, Batch size = 192, early stopping patience = 10	Training time	344. 24 sec
		RMSE	21.836
		MAE	13.183
		R2	0.945
		NSE	0.945

Figure 4.5.3 and Figure 4.5.4 show the loss curves of baseline Bi LSTM model and Best BiLSTM model respectively.

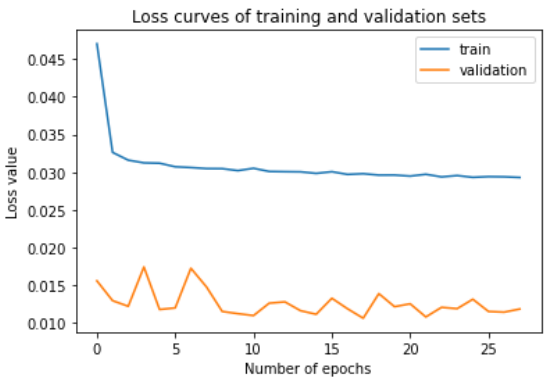


Figure 4.5.3: Baseline BiLSTM.

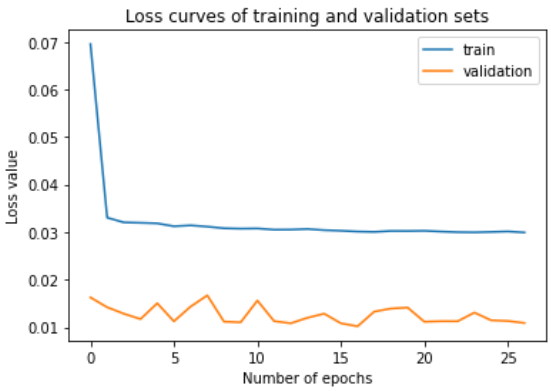


Figure 4.5.4: Best BiLSTM.

Comparing Figure 4.5.3 and Figure 4.5.4, the loss curves of both training and validation sets for both the baseline BiLSTM model and the Best BiLSTM model are not greatly different. This was also clear from the small difference in test RMSE scores of both. So, unlike LSTM, BiLSTM regularization was improved through hyperparameter tuning but the performance of the model hasn't changed much compared to the baseline model.

4.5.3 GRU

The results of GRU models' performance for test data, starting from the baseline model to the best model are mentioned in Table 4.5.5. Note that the output layer was a dense layer of one neuron for all models which is why it is not mentioned anywhere. All the layers in Table 4.5.5 are of GRU.

Table 4.5.5: Results of GRU model in progression from baseline to best model.

Model	***Tunning method***	***Model parameters***	***Test RMSE***
Model0 (baseline)	-	1 layer, 64 neurons, learning rate = 0.01	23.880
Model11	Hyperband	1 layer, 150 neurons, learning rate = 0.01	23.088
Model12	Bayesian optimization	1 layer, 50 neurons, learning rate = 0.01	52.261
Model13	Hyperband	1 layer, 80 neurons, learning rate = 0.01	20.096
Model14	Bayesian optimization	1 layer, 80 neurons, learning rate = 0.0001	23.651
Model15	Hyperband	1 layer, 85 neurons, learning rate = 0.01	28.543
Model16	Bayesian optimization	1 layer, 80 neurons, learning rate = 0.001	21.506
Model21	Hyperband	Layer 1, 70 neurons, Layer 2, 70 neurons, learning rate = 0.001	23.445
Model22	Bayesian optimization	Layer 1, 50 neurons, Layer 2, 80 neurons, Layer 3, 50 neurons, learning rate = 0.0001	21.302
Model17	-	1 layer, 80 neurons, learning rate = 0.01, activation = ‘sigmoid’	22.113
Model18	-	1 layer, 80 neurons, learning rate = 0.01, activation = ‘relu’	22.105
Model31	Hyperband	1 layer, 80 neurons, learning rate = 0.01, Dropout = 0.5	26.550
Model32	Bayesian optimization	1 layer, 80 neurons, learning rate = 0.01, Dropout = 0.8	35.363
Model41	Hyperband	1 layer, 80 neurons, learning rate = 0.01, Dropout = 0.5, weight decay = 0.0001	33.915

Model42	Bayesian optimization	1 layer, 80 neurons, learning rate = 0.01, Dropout = 0.5, weight decay = 0.00001	22.778
Model51	-	1 layer, 80 neurons, learning rate = 0.01, Dropout = 0.5, weight decay = 0.00001, Batch size = 32	40.800
Model52	-	1 layer, 80 neurons, learning rate = 0.01, Dropout = 0.5, weight decay = 0.00001, Batch size = 128	27.954
Model53	-	1 layer, 80 neurons, learning rate = 0.01, Dropout = 0.5, weight decay = 0.00001, Batch size = 192	29.082

After training and testing the baseline model (Model0), hyperparameter tuning was done using Hyperband as well as Bayesian-Optimization methods to search for an optimal number of neurons and an optimal value of learning rate for a single layer GRU. For this, all models from Model11 to Model16 were trained and tested. Then, a search was done for an optimal number of layers, an optimal number of neurons and an optimal value of learning rate for a multilayer GRU, through Model21 and Model 22, which indicated that multilayer GRUs haven't performed better than a single layer. Based on test RMSE scores of all models from Model11 to Model22, Model13 with a single layer of 80 neurons and a learning rate of 0.01 gave the best performance. So, the parameters of this model were considered for the best model.

Next, a comparison of the performance of Model17 and Model18 with Model13 showed that the default activation of tanh was the optimal activation function. From the results of Model31 and Model32, it was concluded that a dropout value of 0.5 was optimal to regularize the model. Similarly, from the results of Model41 and Model42, a weight decay value of 0.00001 was optimal to further regularize the model. Finally, the performance of Model51, Model52 and Model53 indicated that training the models with a batch size of 64 gave the best performance. So, all these parameters were considered for the best model.

With all the optimal parameters obtained above, the best GRU model was created, trained, tested and evaluated. The details of the best GRU parameters and performance metrics are described in Table 4.5.6.

Table 4.5.6: Best GRU parameters and Performance metrics.

Model	***Parameters***	***Evaluation metric***	***Value/ Score***
Best GRU	Number of layers = 1, number of neurons = 80,	Training time	137. 31 sec

	Dropout = 0.5, weight decay = 0.00001, Learning rate = 0.01, epochs = 100, Batch size = 64, early stopping patience = 10	RMSE	22.778
		MAE	14.557
		R2	0.940
		NSE	0.940

Figure 4.5.5 and Figure 4.5.6 show the loss curves of the baseline GRU model and the Best GRU model respectively.

Figure 4.5.5: Baseline GRU.

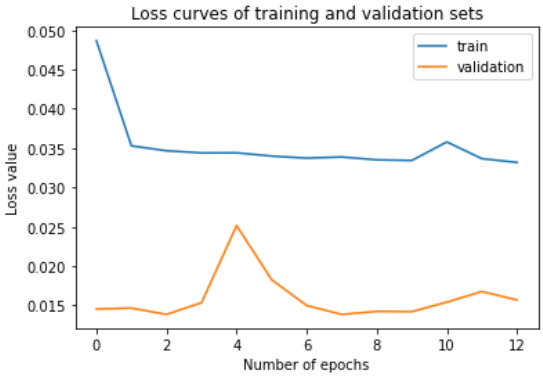


Figure 4.5.6: Best GRU.

Comparing Table 4.5.5 and Table 4.5.6, the loss curves of training sets for both baseline GRU and Best GRU models are very similar, with a little smoothing effect on the latter. On the other hand, the loss curve for the validation set of the Best GRU model is very smooth compared to that of the baseline GRU model.

4.5.4 Bidirectional GRU or BiGRU

The results of BiGRU models' performance for test data, starting from the baseline model to the best model are mentioned in Table 4.5.7. Note that the output layer was a dense layer of one neuron for all models which is why it is not mentioned anywhere. All the layers in Table 4.5.7 are of BiGRU.

Table 4.5.7: Results of BiGRU model in progression from baseline to best model.

Model	***Tunning method***	***Model parameters***	***Test RMSE***
Model0 (baseline)	-	1 layer, 64 neurons, learning rate = 0.01	23.919
Model11	Hyperband	1 layer, 200 neurons, learning rate = 0.001	22.746
Model12	Bayesian optimization	1 layer, 200 neurons, learning rate = 0.01	22.812
Model13	Hyperband	1 layer, 80 neurons, learning rate = 0.0001	20.674
Model14	Bayesian optimization	1 layer, 100 neurons, learning rate = 0.0001	22.944
Model15	Hyperband	1 layer, 80 neurons, learning rate = 0.001	22.122
Model16	Bayesian optimization	1 layer, 70 neurons, learning rate = 0.0001	21.991
Model21	Hyperband	Layer 1, 90 neurons, Layer 2, 80 neurons, learning rate = 0.0001	22.275
Model22	Bayesian optimization	Layer 1, 90 neurons, Layer 2, 70 neurons, Layer 3, 60 neurons, Layer 4, 60 neurons, learning rate = 0.0001	24.771
Model23	Hyperband	Layer 1, 85 neurons, Layer 2, 95 neurons, learning rate = 0.0001	22.226
Model24	Bayesian optimization	Layer 1, 95 neurons, Layer 2, 80 neurons, learning rate = 0.01	37.730
Model17	-	1 layer, 80 neurons, learning rate = 0.0001, activation = ‘sigmoid’	24.204
Model18	-	1 layer, 80 neurons, learning rate = 0.0001, activation = ‘relu’	21.303
Model31	Hyperband	1 layer, 80 neurons, learning rate = 0.0001, Dropout = 0.2	21.362
Model32	Bayesian optimization	1 layer, 80 neurons, learning rate = 0.0001, Dropout = 0.9	28.777

Model41	Hyperband	1 layer, 80 neurons, learning rate = 0.0001, Dropout = 0.2, weight decay = 0.0001	26.316
Model42	Bayesian optimization	1 layer, 80 neurons, learning rate = 0.0001, Dropout = 0.2, weight decay = 0.001	43.531
Model51	-	1 layer, 80 neurons, learning rate = 0.0001, Dropout = 0.2, weight decay = 0.0001, Batch size = 32	23.861
Model52	-	1 layer, 80 neurons, learning rate = 0.0001, Dropout = 0.2, weight decay = 0.0001, Batch size = 128	24.276
Model53	-	1 layer, 80 neurons, learning rate = 0.0001, Dropout = 0.2, weight decay = 0.0001, Batch size = 192	24.168

After training and testing the baseline model (Model0), hyperparameter tuning was done using Hyperband as well as Bayesian-Optimization methods to search for an optimal number of neurons and an optimal value of learning rate for a single layer BiGRU. For this, all models from Model11 to Model16 were trained and tested. Then, a search was done for an optimal number of layers, an optimal number of neurons and an optimal value of learning rate for a multilayer BiGRU, from Model21 to Model 24. Based on test RMSE scores of all models from Model11 to Model24, Model13 with a single layer of 80 neurons and a learning rate of 0.0001 gave the best performance. So, the parameters of this model were considered for the best model.

Next, a comparison of the performance of Model17 and Model18 with Model13 showed that the default activation of tanh was the optimal activation function. From the results of Model31 and Model32, it was concluded that a dropout value of 0.2 was optimal to regularization the model. Similarly, from the results of Model41 and Model42, a weight decay value of 0.0001 was optimal to further regularize the model. Finally, the performance of Model51, Model52 and Model53 indicated that training the models with a batch size of 32 gave the best performance. So, all these parameters were considered for the best model.

With all the optimal parameters obtained above, the best BiGRU model was created, trained, tested and evaluated. The details of the best BiGRU parameters and performance metrics are described in Table 4.5.8.

Table 4.5.8: Best BiGRU parameters and Performance metrics.

Model	***Parameters***	***Evaluation metric***	***Value/ Score***

Best BiGRU	Number of layers = 1, number of neurons = 80, Dropout = 0.2, weight decay = 0.0001, Learning rate = 0.0001, epochs = 100, Batch size = 32, early stopping patience = 10	Training time	879. 06 sec
		RMSE	24.601
		MAE	14.548
		R2	0.930
		NSE	0.930

Figure 4.5.7 and Figure 4.5.8 show the loss curves of baseline BiGRU model and Best BiGRU model respectively.

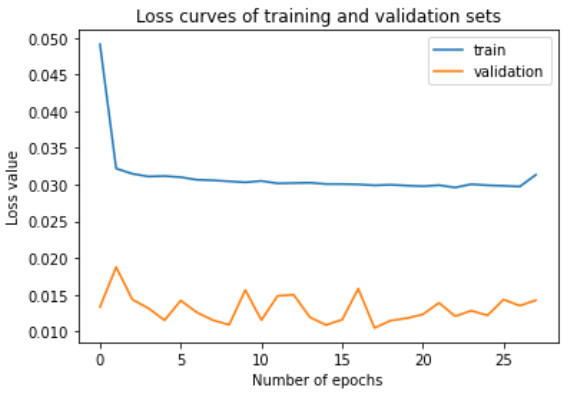


Figure 4.5.7: Baseline BiGRU.

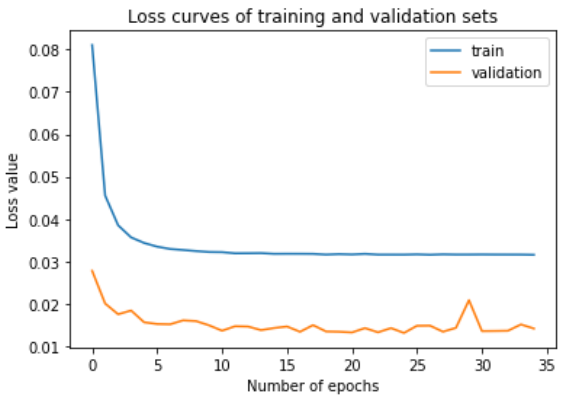


Figure 4.5.8: Best BiGRU.

Comparing Figure 4.5.7 and Figure 4.5.8, the loss curves of training sets and validation sets of both baseline BiGRU and Best BiGRU, it can be clearly said that the curves of the latter are smoother than that of the former, which is obviously due to tuning and regularization.

Table 4.5.9 provides a comparison of the performances of the best models of all four algorithms used in the current project.

Table 4.5.9: Comparison of Best models of all four algorithms.

Metric	***LSTM***	***BiLSTM***	***GRU***	***BiGRU***
Training time	293. 76 sec	344. 24 sec	137. 31 sec	879. 06 sec
RMSE	24.491	21.836	22.778	24.601
MAE	15.149	13.183	14.557	14.548
R2	0.931	0.945	0.940	0.930
NSE	0.929	0.945	0.940	0.930

In conclusion, from Table 4.5.9, BiLSTM is the best performing algorithm out of all, followed by GRU. Just because of training time, LSTM is better than BiGRU, which otherwise would have been a tie.

4.6 Evaluation of transfer learning

The evaluation of transfer learning was done with the data of all the cities other than Delhi., i.e., Bengaluru, Hyderabad, Jaipur and Mumbai. To evaluate the impact of transfer learning the performance metrics of the best models of all the algorithms with transfer learning were compared to the corresponding case without transfer learning, for all four cities. In the performance metrics, except for training time, the rest of the metrics are for test data.

4.6.1 For LSTM

Performance of Best LSTM for Bengaluru city was as in Table 4.6.1.

Table 4.6.1: Performance of Best LSTM – Bengaluru city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	94.79 sec	190.29 sec
RMSE	3.954	4.055
MAE	2.499	2.489
R2	0.528	0.504
NSE	0.528	0.504

As per Table 4.6.1, transfer learning was a clear winner in terms of computational efficiency. In terms of error values, TL performed better for RMSE but not for MAE. In terms of accuracy metrics, TL performed better in both R2 and NSE.

The loss curves of these two cases are as shown in Figure 4.6.1 which indicate smoother curves in the case of transfer learning meaning better training of the model.

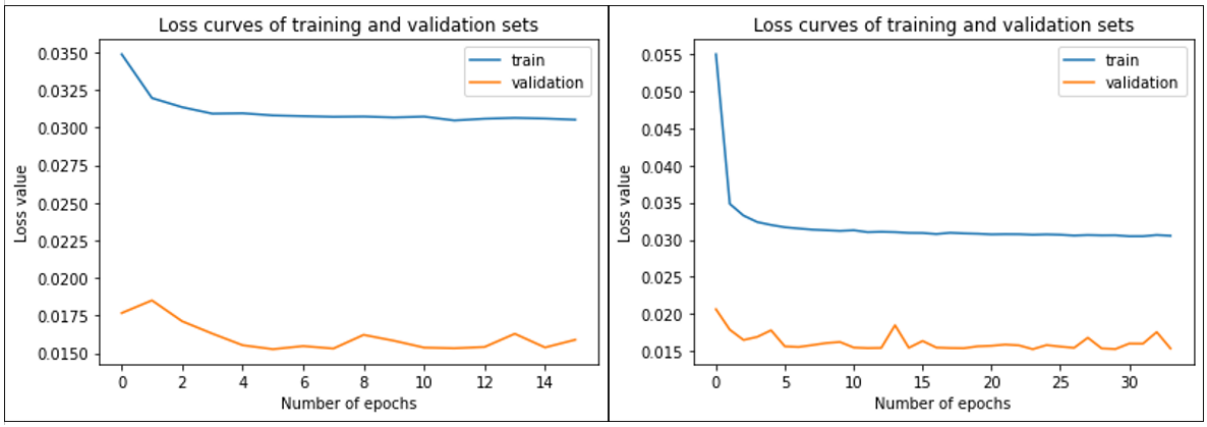


Figure 4.6.1: Best LSTM – Bengaluru (left – with TL, right – no TL).

So, for Bengaluru overall performance of LSTM with TL was slightly better.

Performance of Best LSTM for Hyderabad city was as in Table 4.6.2.

Table 4.6.2: Performance of Best LSTM – Hyderabad city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	137.65 sec	122.41 sec
RMSE	2.964	2.855
MAE	1.910	1.743
R2	0.712	0.733
NSE	0.712	0.733

As per Table 4.6.2, transfer learning performed poorly than without transfer learning in all the metrics.

The loss curves of these two cases are as shown in Figure 4.6.2 which indicate a smoother curve in the case without transfer learning.

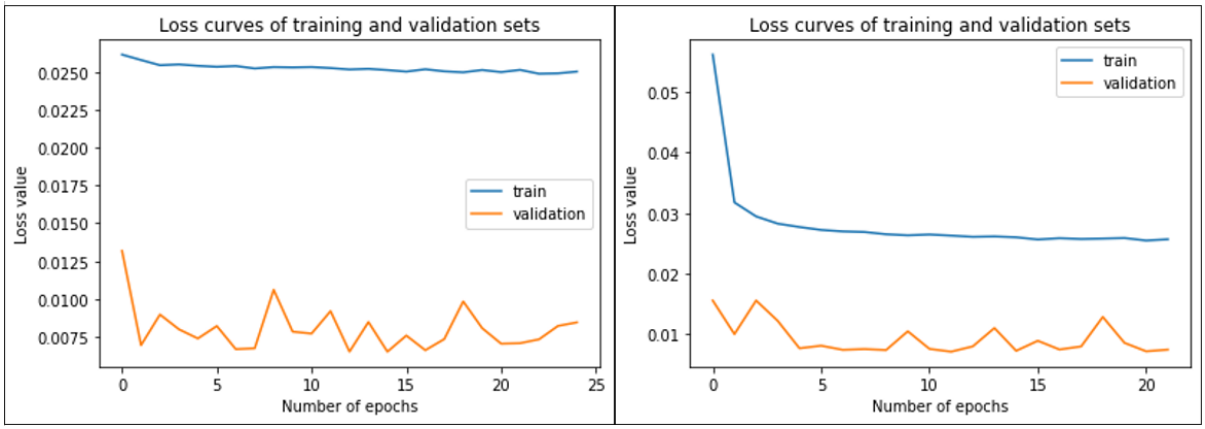


Figure 4.6.2: Best LSTM – Hyderabad (left – with TL, right – no TL).

So, for Hyderabad LSTM with TL wasn't better at all.

Performance of Best LSTM for Jaipur city was as in Table 3.1.1.

Table 4.6.3: Performance of Best LSTM – Jaipur city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	68.61 sec	120.79 sec
RMSE	3.201	3.116
MAE	1.409	1.444
R2	0.726	0.740
NSE	0.726	0.740

As per Table 4.6.3, although TL performed better in training time and MAE, it was poor in the rest of the metrics.

The loss curves of these two cases are as shown in Figure 4.6.3 which indicate a smoother curve in cases without transfer learning.

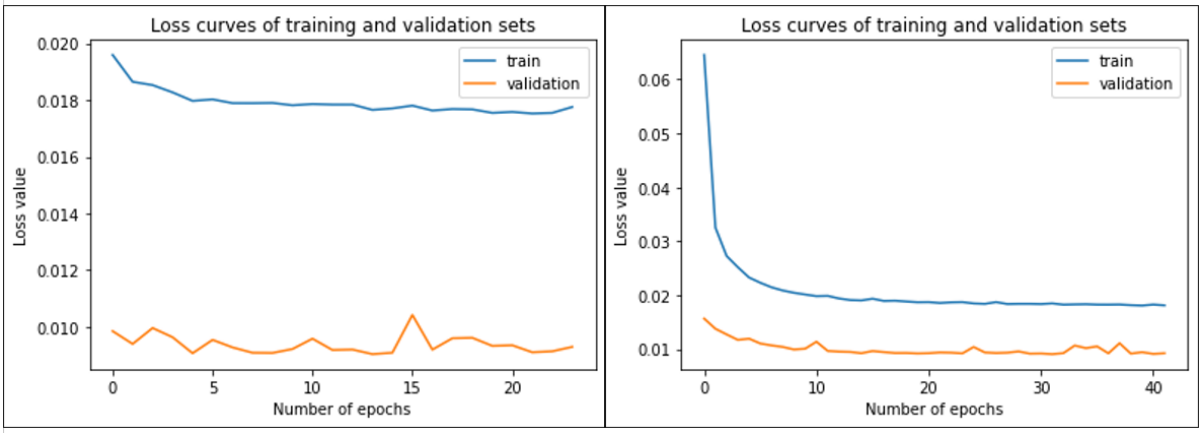


Figure 4.6.3: Best LSTM – Jaipur (left – with TL, right – no TL).

So, for Jaipur for LSTM with TL wasn't better.

Performance of Best LSTM for Mumbai city was as in Table 4.6.4.

Table 4.6.4: Performance of Best LSTM – Mumbai city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	60.14 sec	100.51 sec
RMSE	3.472	3.513
MAE	2.112	2.309
R2	0.691	0.684

NSE	0.691	0.684

As per Table 4.6.4, TL performed better in all the metrics.

The loss curves of these two cases are as shown in Figure 4.6.4 which indicate a smoother and more stable curve in the case of transfer learning.

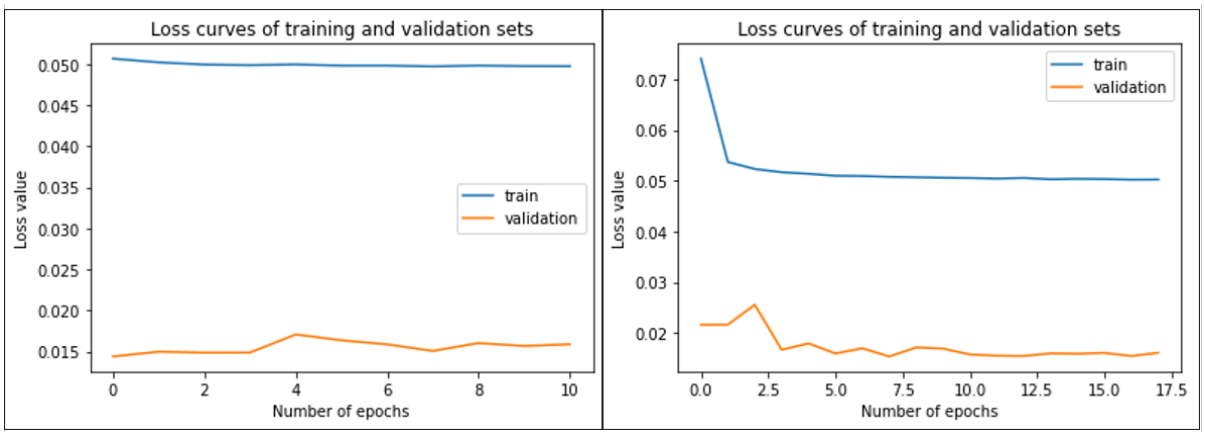


Figure 4.6.4: Best LSTM – Mumbai (left – with TL, right – no TL).

So, for Mumbai LSTM with TL performed better every way.

4.6.2 For Bidirectional LSTM or BiLSTM

Performance of Best BiLSTM for Bengaluru city was as in Table 4.6.5.

Table 4.6.5: Performance of Best BiLSTM – Bengaluru city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	187.88 sec	212.23 sec
RMSE	4.135	4.133
MAE	2.893	2.494
R2	0.484	0.484
NSE	0.484	0.484

As per Table 4.6.5, TL was better only in terms of training time, but not in other metrics.

The loss curves of these two cases are shown in Figure 4.6.5 which indicates a smoother curve in the case of transfer learning.

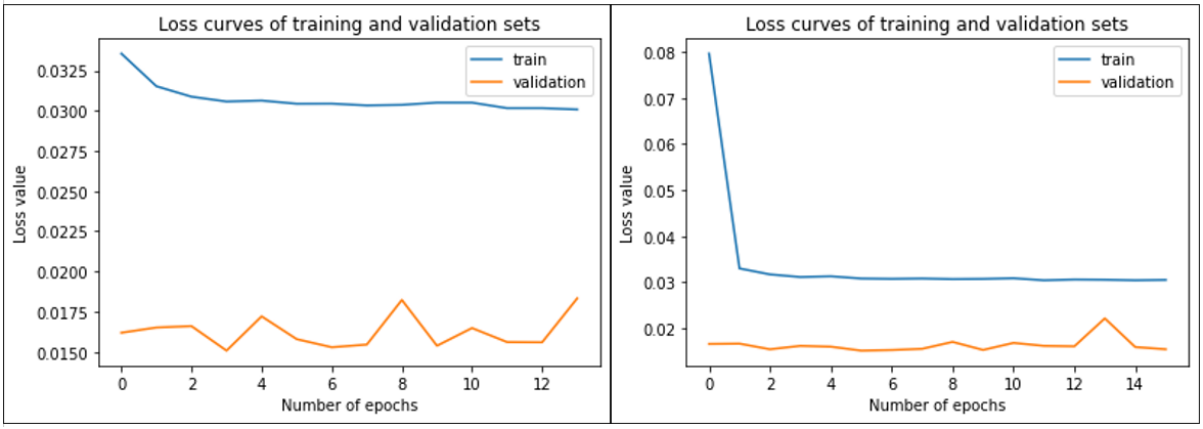


Figure 4.6.5: Best BiLSTM – Bengaluru (left – with TL, right – no TL).

So, for Bengaluru BiLSTM with TL didn't perform better in overall.

Performance of Best BiLSTM for Hyderabad city was as in Table 4.6.6.

Table 4.6.6: Performance of Best BiLSTM – Hyderabad city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	250.70 sec	256.90 sec
RMSE	2.785	2.913
MAE	1.661	1.840
R2	0.746	0.722
NSE	0.746	0.722

As per Table 4.6.6, transfer learning performed better than without transfer learning in all the metrics.

The loss curves of these two cases are as shown in Figure 4.6.6 which indicate a smoother curve strangely in cases without transfer learning, contrary to metrics results.

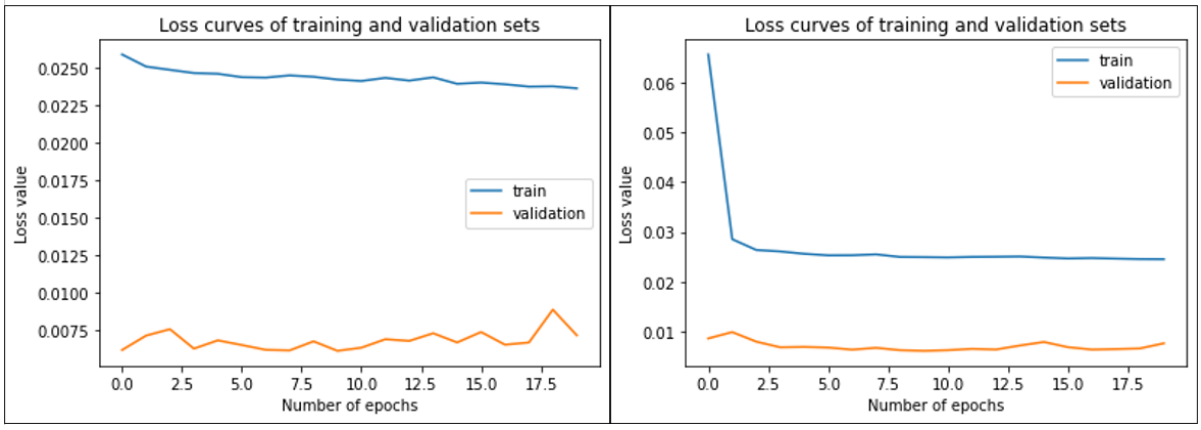


Figure 4.6.6: Best BiLSTM – Hyderabad (left – with TL, right – no TL).

So, for Hyderabad BiLSTM with TL was better in all metrics.

Performance of Best BiLSTM for Jaipur city was as in Table 4.6.7.

Table 4.6.7: Performance of Best BiLSTM – Jaipur city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	191.20 sec	166.00 sec
RMSE	3.799	3.377
MAE	1.611	1.753
R2	0.614	0.695
NSE	0.614	0.695

As per Table 4.6.7, although TL didn't perform better in training time and RMSE, it was better in the rest of the metrics.

The loss curves of these two cases are as shown in Figure 4.6.7 which indicate the curves to be almost similar for both cases.

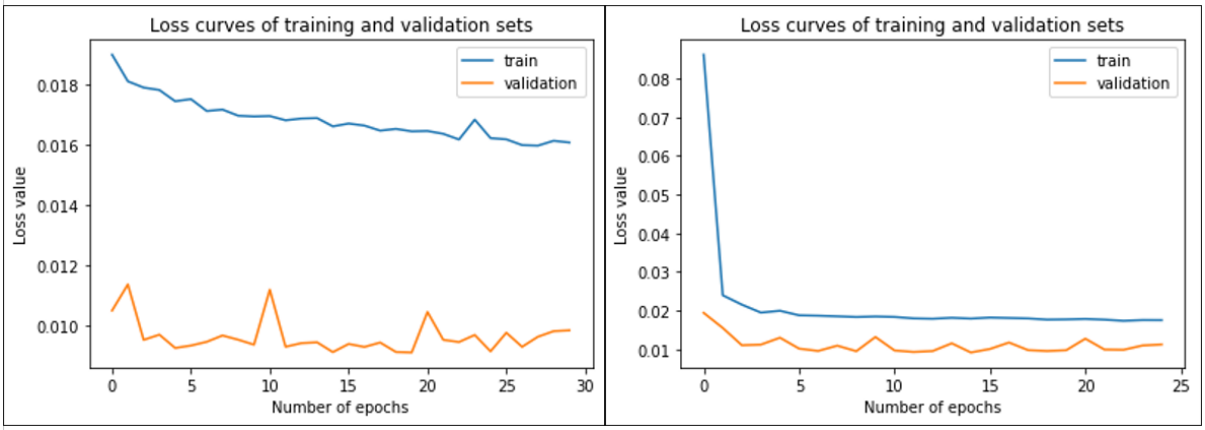


Figure 4.6.7: Best BiLSTM – Jaipur (left – with TL, right – no TL).

So, for Jaipur BiLSTM with TL was better in overall.

Performance of Best BiLSTM for Mumbai city was as in Table 4.6.8.

Table 4.6.8: Performance of Best BiLSTM – Mumbai city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	137.60 sec	358.93 sec
RMSE	3.459	3.632
MAE	2.155	2.445
R2	0.694	0.662

NSE	0.694	0.662

As per Table 4.6.8, TL performed better in all the metrics.

The loss curves of these two cases are as shown in Figure 4.6.8 which indicate a smoother and more stable curve in the case of transfer learning.

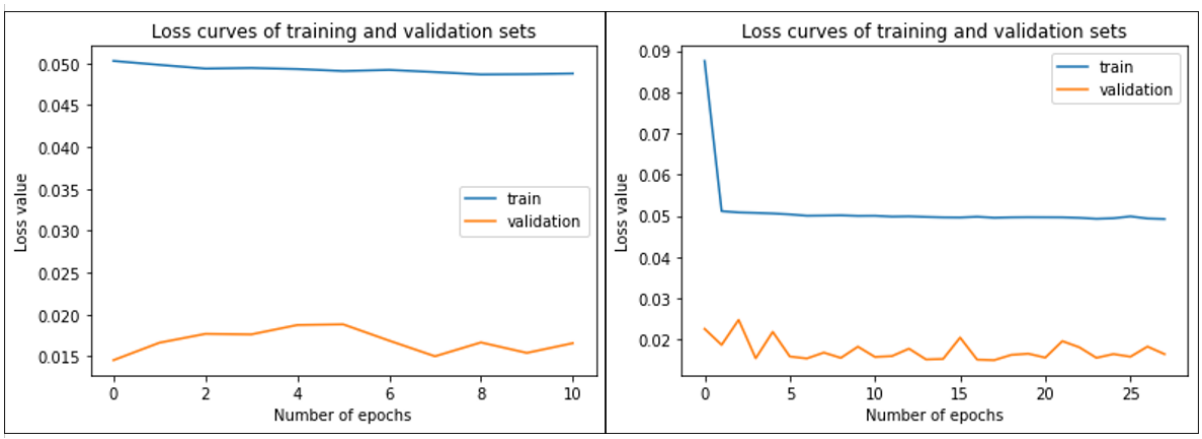


Figure 4.6.8: Best BiLSTM – Mumbai (left – with TL, right – no TL).

So, for Mumbai BiLSTM with TL performed better in every way.

4.6.3 For GRU

Performance of Best GRU for Bengaluru city was as in Table 4.6.9.

Table 4.6.9: Performance of Best GRU – Bengaluru city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	229.46 sec	250.43 sec
RMSE	4.047	4.650
MAE	2.456	3.200
R2	0.506	0.347
NSE	0.506	0.347

As per Table 4.6.9, TL was better in all the metrics.

The loss curves of these two cases are as shown in Figure 4.6.9 which indicate a smoother and more stable curve in the case of transfer learning, achieving a proper convergence in training.

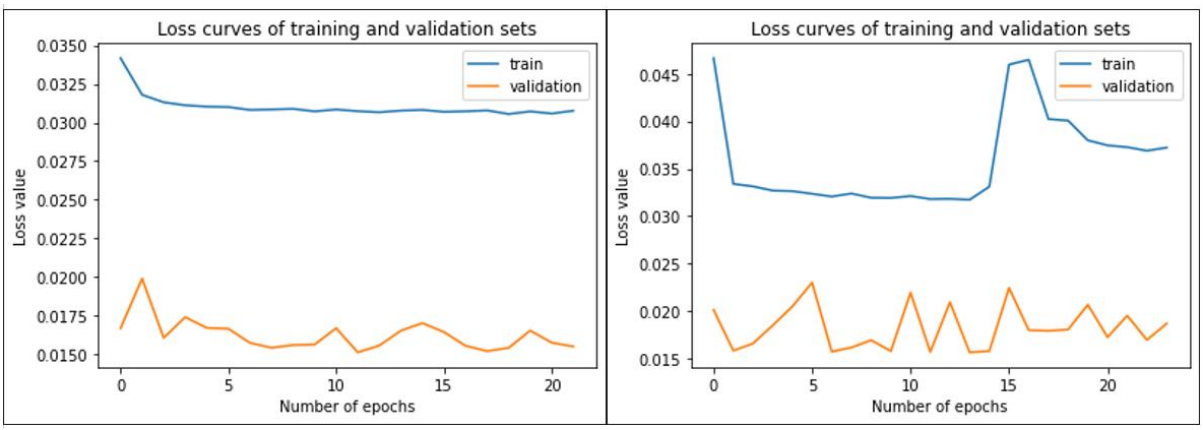


Figure 4.6.9: Best GRU – Bengaluru (left – with TL, right – no TL).

So, for Bengaluru GRU with TL performed better in every way.

Performance of Best GRU for Hyderabad city was as in Table 4.6.10.

Table 4.6.10: Performance of Best GRU – Hyderabad city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	163.31 sec	237.74 sec
RMSE	2.712	5.205
MAE	1.587	4.317
R2	0.759	0.112
NSE	0.759	0.112

As per Table 4.6.10, transfer learning performed better than without transfer learning in all the metrics.

The loss curves of these two cases are as shown in Figure 4.6.10 which indicate smoother curves in the case of transfer learning to a very large extent.

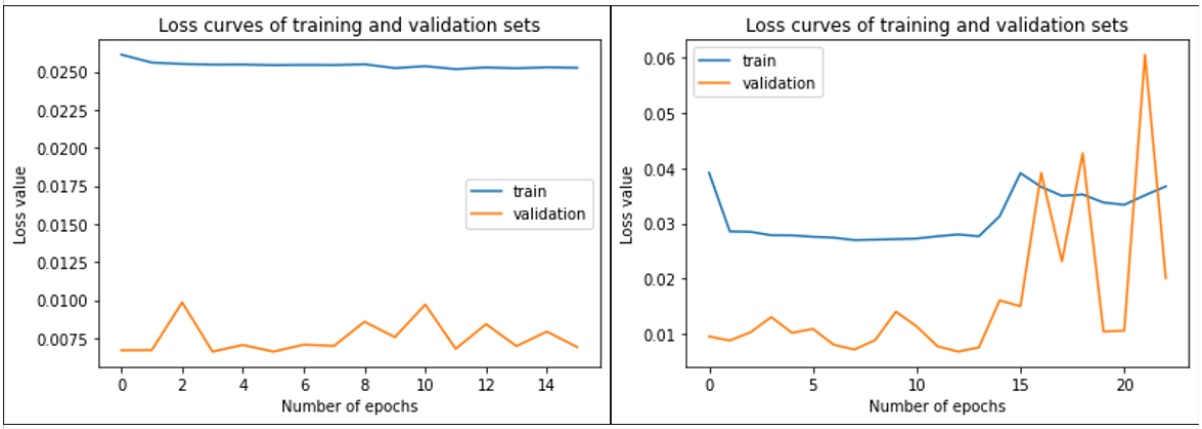


Figure 4.6.10: Best GRU – Hyderabad (left – with TL, right – no TL).

So, for Hyderabad GRU with TL was better in every way.

Performance of Best GRU for Jaipur city was as in Table 4.6.11.

Table 4.6.11: Performance of Best GRU – Jaipur city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	158.25 sec	107.16 sec
RMSE	3.232	3.395
MAE	1.502	1.967
R2	0.720	0.691
NSE	0.720	0.691

As per Table 4.6.11, although TL didn't perform better in training time, it was better in the rest of the metrics.

The loss curves of these two cases are as shown in Figure 4.6.11 which indicate a smoother curve in the case of transfer learning to a very large extent.

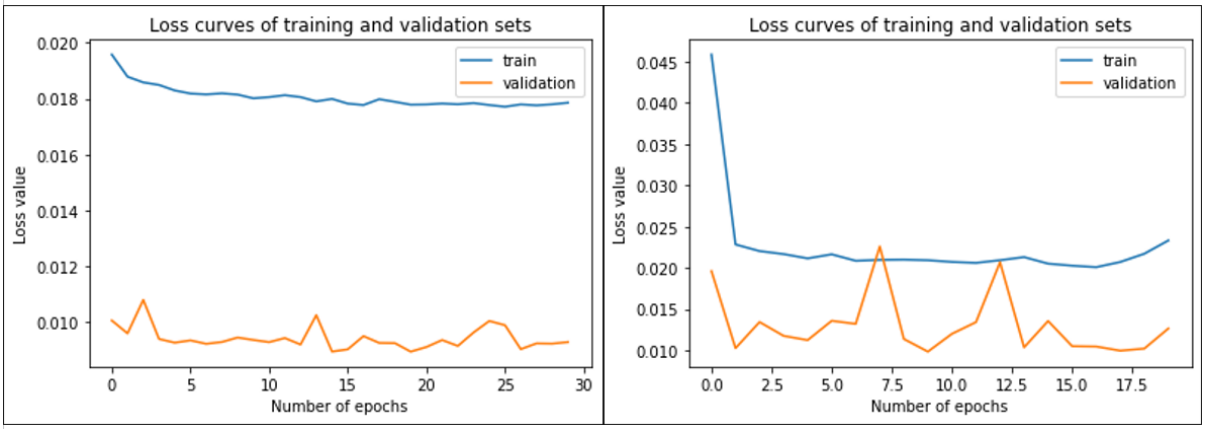


Figure 4.6.11: Best GRU – Jaipur (left – with TL, right – no TL).

So, for Jaipur GRU with TL was better in overall.

Performance of Best GRU for Mumbai city was as in Table 4.6.12.

Table 4.6.12: Performance of Best GRU – Mumbai city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	142.75 sec	206.57 sec
RMSE	3.503	3.962
MAE	2.413	2.582
R2	0.686	0.598

NSE	0.686	0.598

As per Table 4.6.12, TL performed better in all the metrics.

The loss curves of these two cases are as shown in Figure 4.6.12 which indicate a smoother and more stable curve in the case of transfer learning.

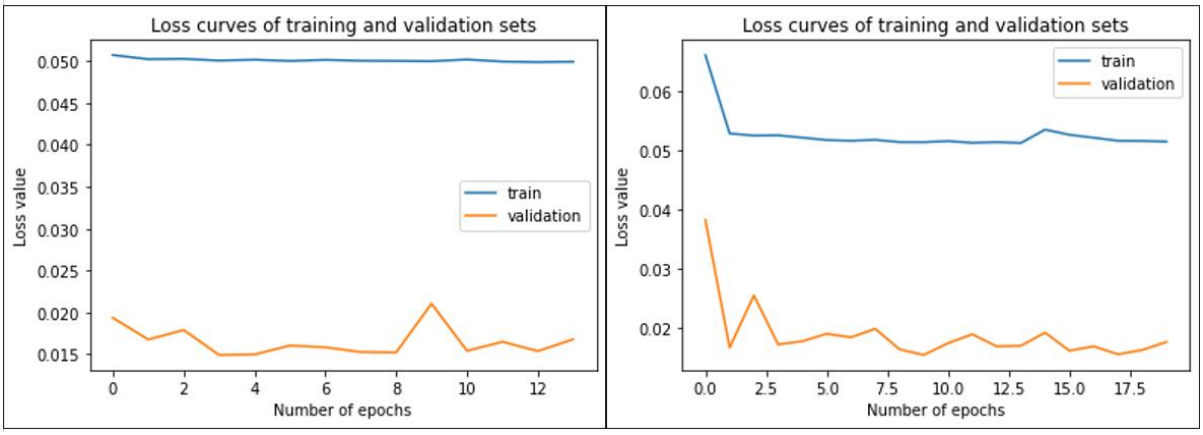


Figure 4.6.12: Best GRU – Mumbai (left – with TL, right – no TL).

So, for Mumbai GRU with TL performed better in every way.

4.6.4 For Bidirectional GRU or BiGRU

Performance of Best BiGRU for Bengaluru city was as in Table 4.6.13.

Table 4.6.13: Performance of Best BiGRU – Bengaluru city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	949.73 sec	796.80 sec
RMSE	4.111	4.254
MAE	2.535	2.677
R2	0.490	0.454
NSE	0.490	0.454

As per Table 4.6.13, except for training time, TL was better in all the metrics.

The loss curves of these two cases are as shown in Figure 4.6.13 which indicate a smoother curve strangely in cases without transfer learning, contrary to metrics results.

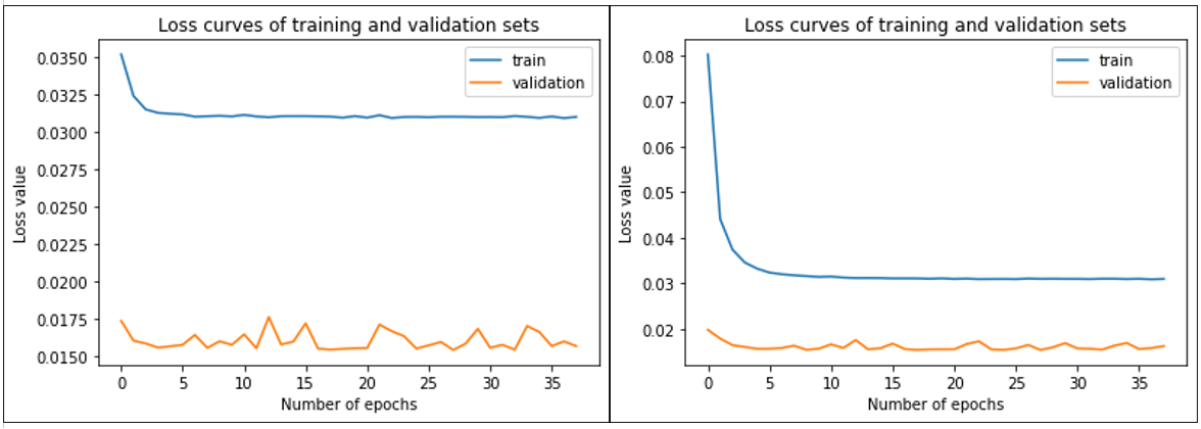


Figure 4.6.13: Best BiGRU – Bengaluru (left – with TL, right – no TL).

So, for Bengaluru BiGRU with TL performed better in overall.

Performance of Best BiGRU for Hyderabad city was as in Table 4.6.14.

Table 4.6.14: Performance of Best BiGRU – Hyderabad city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	417.46 sec	639.32 sec
RMSE	2.563	2.545
MAE	1.514	1.540
R2	0.785	0.788
NSE	0.785	0.788

As per Table 4.6.14, TL perform better only in training time and MAE.

The loss curves of these two cases are as shown in Figure 4.6.14 which indicate that curves are similarly smooth for both TL and no TL.

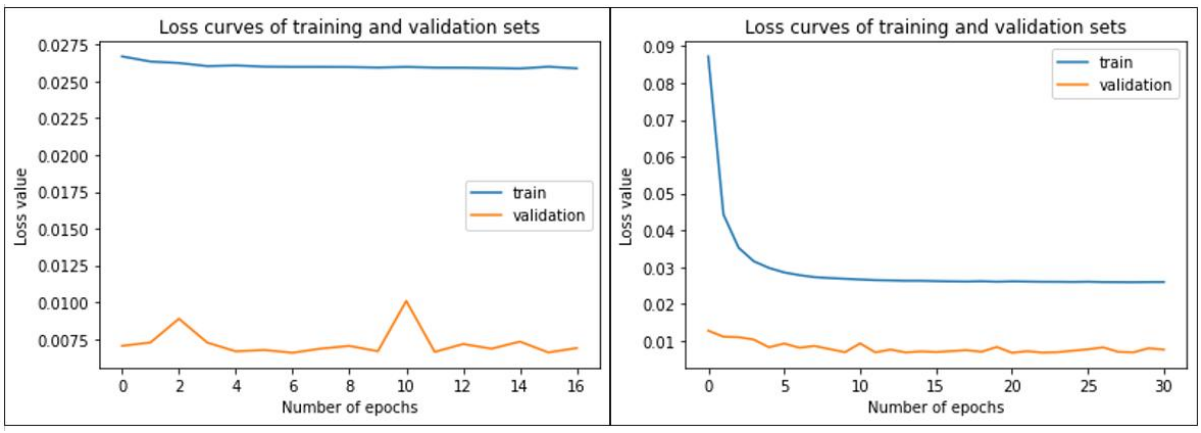


Figure 4.6.14: Best BiGRU – Hyderabad (left – with TL, right – no TL).

So, for Hyderabad BiGRU with TL wasn't better in overall.

Performance of Best BiGRU for Jaipur city was as in Table 4.6.15.

Table 4.6.15: Performance of Best BiGRU – Jaipur city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	746.93 sec	548.48 sec
RMSE	3.131	3.103
MAE	1.497	1.457
R2	0.738	0.742
NSE	0.738	0.742

As per Table 4.6.15, TL didn't perform better in any of the metrics.

The loss curves of these two cases are as shown in Figure 4.6.15 which indicate a smoother curve in cases without transfer learning.

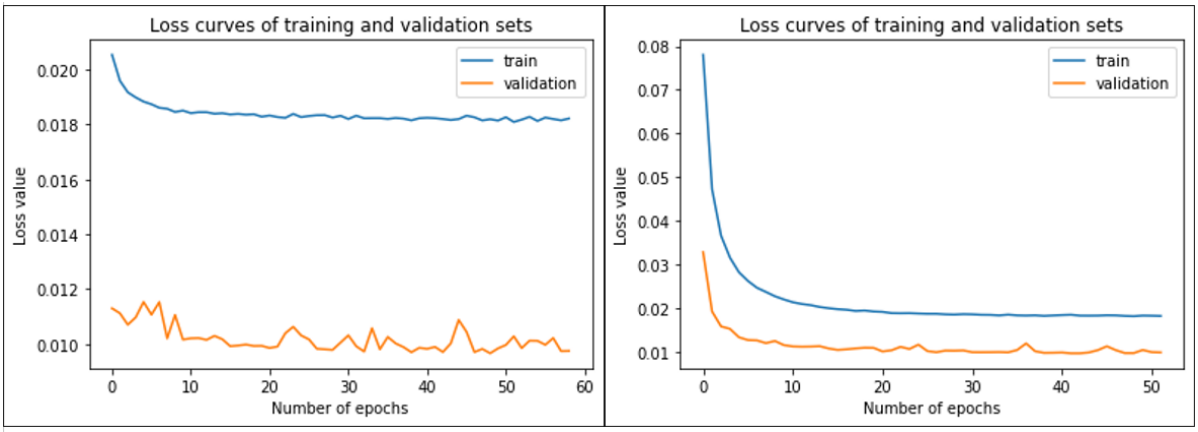


Figure 4.6.15: Best BiGRU – Jaipur (left – with TL, right – no TL).

So, for Jaipur BiGRU with TL wasn't better in any way.

Performance of Best BiGRU for Mumbai city was as in Table 4.6.16.

Table 4.6.16: Performance of Best BiGRU – Mumbai city.

Metric	***With transfer learning (TL)***	***Without transfer learning (no TL)***
Training time	368.45 sec	868.01 sec
RMSE	3.596	3.543
MAE	2.310	2.227
R2	0.669	0.678
NSE	0.669	0.678

As per Table 4.6.16, except for training time, TL didn't perform better in any of the metrics.

The loss curves of these two cases are as shown in Figure 4.6.16 which indicate that curves are similarly smooth for both TL and no TL.

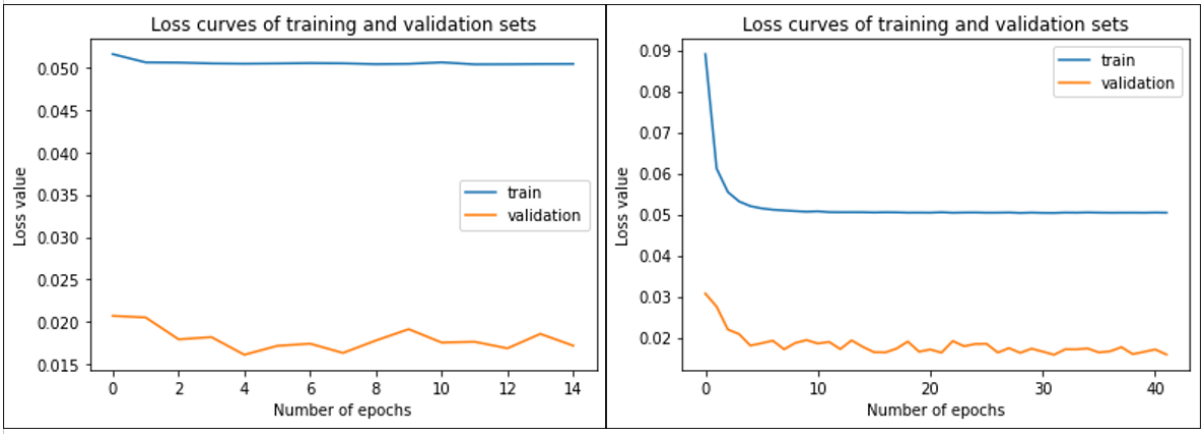


Figure 4.6.16: Best BiGRU – Mumbai (left – with TL, right – no TL).

So, for Mumbai BiGRU with TL didn't perform better in overall.

Table 4.6.17 summarizes all the 16 cases above, out of transfer learning (TL) and without transfer learning (no TL), which case was better.

Table 4.6.17: Summary of TL vs no TL for all the 16 cases above.

Model	**Bengaluru**	**Hyderabad**	**Jaipur**	**Mumbai**
Best LSTM	TL	no TL	no TL	TL
Best BiLSTM	no TL	TL	TL	TL
Best GRU	TL	TL	TL	TL
Best BiGRU	TL	no TL	no TL	no TL

4.7 Conclusion

From all the above sections of this chapter, it can be concluded that the results of every section played a vital role in obtaining the results for its subsequent sections.

Quantitatively, Table 4.6.17 shows that in 10 out of 16 cases, transfer learning improved the performance of the models, although to a very small extent. This approximates 62.5% of cases, which is an answer to the research question of the current project.

Answer to the research question:

"The conclusion of results of the current project shows that in about 62.5% cases, even though to a very small extent, transfer learning has improved the performance of RNNs on the data of cities with a lower variance of PM2.5 levels when the same models were originally trained on the data of a city with higher variance in PM2.5 levels."

Chapter 5 Discussion

5.1 Examination of results with respect to objectives of the current project

The examination of the results with respect to the objectives of the current project will be done objective-wise, one after the other.

5.1.1 Data pre-processing

Objective - To find out techniques to clean the data based on literature review, as per the need of the data and in view of the purpose of the project to prepare the data useable for analysis.

The boxplot of all the columns of air pollution data presented in chapter 4, was the starting point of understanding the distribution of the data, which showed the direction to proceed further with data preprocessing.

One of the first steps in data preprocessing was the imputation of missing values in air pollution data, for which an interpolation technique was applied that interpolated the missing values based on the earlier timestamps. Distribution plots shown in chapter 4, had indicated that this technique practically didn't change the distribution of the data, thereby almost maintaining the statistical nature of the data. This was also confirmed by the descriptive statistics tables, that numerically confirmed no significant changes, especially in mean and standard deviation.

The correlation matrix shown in chapter 4, helped to determine the redundant columns of the data and drop them, which otherwise would have impacted the performance of the model. Similarly, descriptive statistics aided in the process of outliers identification, although no outliers were removed based on those statistics and other reasons.

Hence, the objective was fulfilled.

5.1.2 Exploratory data analysis

Objective – To recognize meaningful patterns among the data, try to understand the underlying relations among the data and to obtain the right set or subset of data that would be suitable for model creation. To know the city for which the variance in PM2.5 level is higher than others.

The histograms of the minimum, average and maximum values of PM2.5 for different cities informed through data analysis that Delhi is the city with higher variance in PM2.5 levels, which is one of the important results given the purpose of the project.

Line plots of average monthly values of PM2.5 and other pollutants uncovered periodicity and seasonality among the emission patterns of the pollutants that implied the importance of meteorological data. They also showed the relationship between PM2.5 and other pollutants. The scatter plots of PM2.5

against other pollutants have confirmed the non-existence of spatial influence on the variance of pollution data and also indicated the correlations between PM2.5 and other pollutants.

Therefore, the objective was achieved.

5.1.3 Modelling

Objective – To identify the appropriate machine learning algorithms to model based on literature review, exploratory data analysis and the purpose of the project. Then create, train, tune and test the models.

Based on literature review and the purpose of the project, four RNNs viz., LSTM, BiLSTM, GRU and BiGRU were identified for use in the project.

The performance of the models for different iterations of tuning them have helped find the optimal parameters like the number of layers, number of neurons, learning rate of the optimizer, dropout and weight decay regularization parameters and batch size. Identification of these improved the performance of models from a baseline to the best levels, in terms of accuracy as well as generalizability.

The performance results of the best models also showed that all four RNNs have performance in a comparable range and hence are good to analyze the impact of transfer learning.

The loss curves in all cases helped in identifying the presence of any overfitting through the smoothness of the curves of the training set and validation set.

So, the objection was accomplished.

5.1.4 Evaluation

Objective – To find the appropriate evaluation metrics from the literature review to evaluate the models and then discuss the critical points of the results of the project. And finally, with all of this done, provide answer to the research question.

The training time evaluation metric showed the computational efficiency of the models, and the results of training times have shown that except for a few cases, transfer learning improved the computational efficiency of the models.

RMSE and MAE are error evaluation metrics, whereas R2 and NSE are accuracy-based metrics. So, there was a balance of loss and accuracy in the evaluation of the models. These helped to understand whether a model with transfer learning performed better or not and this analysis showed that transfer learning improved the performance of models in 62.5 % of cases. This result has answered the research question quantitatively.

Hence, the objective was attained.

5.2 Examination of results with respect to previous studies

5.2.1 Previous air pollution studies

A study by Krishan *et al.* (2019) was one of the closest to the current project, achieving RMSE of 1.84 and NSE of 0.94 for PM2.5 prediction in Delhi with LSTM, whereas the current project achieved RMSE of 24.491 and NSE of 0.93 with LSTM. So, in terms of NSE, the current project could attain almost a similar performance.

Chelani, Gajghate and Hasan (2002) achieved an RMSE of 7.9 for the prediction of PM10 at Jaipur city using ANN. The current project predicted PM2.5 at Jaipur city and achieved the best RMSE scores of 3.103 using BiGRU. Hence, a better forecast was achieved in the current project.

The results of a study by Gu, Li and Meng (2022) for 1 hour ahead forecast of PM2.5 indicated LSTM to perform better than GRU, the results of the current project were contrary to this, with the performance of GRU better than LSTM.

Bekkar *et al.* (2021) used LSTM, BiLSTM, GRU and BiGRU along with many other models for forecasting PM2.5. Their results indicated that the performance of these models in decreasing order was BiLSTM, LSTM, BiGRU, and GRU. Whereas, the result of the current project indicated decreasing order of performance for the same four models as BiLSTM, GRU, LSTM, and BiGRU.

Jeya and Sankari (2020) obtained MAE of 7.53 and RMSE of 9.86 for forecasting PM2.5 using BiLSTM. In comparison, the best MAE and RMSE this study could obtain using BiLSTM were 1.840 and 2.913 respectively. Therefore, the prediction errors were lesser in the current project.

So, clearly, in terms of PM2.5, the current project developed models that are similar in ability to previous studies.

5.2.2 Previous transfer learning studies

The results of a study by Fong *et al.* (2020) showed higher accuracies for stations with lesser data using transfer learning. In the current project, Jaipur city was one such case with the number of observations almost half of that of other cities. From Table 4.6.17 it can be observed that transfer learning could a similar improvement in model performance for BiLSTM and GRU, but not for LSTM and BiGRU.

Ma *et al.* (2020) used transfer learning with BiLSTM for forecasting PM2.5 and other pollutants, and achieved a test RMSE of 12.1073, an improvement of 35.21% compared to without transfer learning. In the current project, the highest RMSE obtained through transfer learning is 4.135, but it is not such a huge improvement as that in the former study.

In an earlier study Ma *et al.* (2019) used transfer learning with LSTM for forecasting PM2.5 and obtained the best RMSE and MAE values of 8.6529 and 6.1840 respectively. Whereas the best RMSE

and MAE obtained by the current project using transfer learning with LSTM are 2.964 and 1.910 respectively.

Hence, the results of the current project are up to the standards of previous studies in terms of performance metrics, however, the improvement brought by transfer learning isn't that high.

5.2.3 Own contribution of researcher

One of the own contributions of the researcher is the use of Hyperband and Bayesian-Optimization techniques for hyperparameter tuning, which were not used by previous studies for forecasting PM2.5, to the best of the knowledge of the researcher. All the results obtained for model performance were achieved through these techniques and as described above the results are in similar standards to that of previous studies that used a different technique. This implies that the contribution by the researcher can be generalized and the implication of this is improvement in computational efficiency for hyperparameter tuning of RNNs.

The second own contribution of the researcher is the use of the AdamW optimizer, which adds an additional model regularization parameter to the model, improving its generalizability, the implication of this contribution. The validity and generalizability of this are again confirmed by the results of the study.

The third and major own contribution of the researcher is the use of the concept of training models on data of the city with a higher variance of PM2.5 concentrations and then using transfer learning on those models while predicting PM2.5 in other cities with a lower variance of PM2.5 concentration. The implication of this contribution is quantitative proof that in more than half of such cases transfer learning improves model performance for forecasting. The validity and generalizability are again confirmed by the results of the study.

Chapter 6 Evaluation, Reflections, and Conclusions

6.1 Choice of Objectives

The objectives of the project were a little hard to achieve as there were too much of experiments that needed to be done to achieve them and answer the research question, given the limited time to complete the project. Especially, the use of transfer learning over four different cities' data, over and above tuning four RNNs to their best ability was too much for a short-span project.

However, the project could be completed despite some hurdles with the implementation of methods and the answer to the research question was provided with the conclusion of the results of the project.

Although the answer to the research question quantitatively informs that transfer learning can improve the performance of RNNs when predicting PM2.5 for data from cities with lower variance in PM2.5 levels, when the models were trained on the data of the city with higher variance in PM2.5 levels, the answer also mentions that the improvement is only to a slight extent. Increasing the extent of improvement is beyond the scope of the current project.

A major deviation from the original project proposal was the idea to use transfer learning with RNNs. Consequently, the research question and the objectives got modified to a tougher level.

6.2 Literature review examined

The literature review was helpful in many ways. It showed a direction to proceed with all the objectives of the project. Except for the own contributions of the researcher mentioned in chapter 5, most of the methods were inspired by the literature review. Even for own contributions, the literature review served as a guiding light. However, it was difficult to find literature in specific contexts like transfer learning in air pollution prediction and in time series forecasting in general, the concepts are relatively new.

6.3 Methods used

All the methods used made a great contribution to the faster implementation of the project. Considering the time available to complete the project, any other methods would have made it hectic for the researcher, to the best of the knowledge of the researcher.

6.4 Reflection

There were some general conclusions reached by the researcher while carrying out the current project.

Firstly, for the prediction of PM2.5, both air pollution data and meteorological data are needed, which was also observed in many studies of literature review.

Secondly, hyperband and Bayesian optimization techniques for hyperparameter tuning of RNNs are faster than grid search or random search.

Lastly, transfer learning doesn't improve model performance in all cases and in some cases, it may give a poor performance than the case without transfer learning.

The project shows that further experimentation in this direction would help the prediction of PM2.5 in places where there is poor monitoring infrastructure.

6.5 Lessons learnt from the project

Right from the start, the literature review must be done not just for acquiring knowledge, but also to understand the specifics of the research area and be adaptable to change and upgrade the research question.

Keep a regular check on experiments to ensure that any task carried out is linked to one of the objectives of the project.

Always keep the research question in mind to avoid deviating to different tasks as part of experimentation.

6.6 Future work

In future work, there are a few points which can be experimented with. These includes:

1. Better ways to merge the data of different locations or stations than averaging the values like it was done in the current project.
2. Try similar experiments for larger temporal and spatial ranges of data.
3. Try to use transfer learning not just for different cities having similar data, but also for data with fewer similarities.
4. If data is available for large number of cities, then unlike 16 cases in the current project through which the effectiveness of transfer learning was evaluated, with 30 or more cases the evaluation can be validated with statistical tests which would be significant.

6.7 Conclusion

The current project carried out research to predict PM2.5 for a city with a higher variance of PM2.5 levels and developed RNNs by training, tuning and testing them on the data of this city. After that, the same models were just trained and tested on data of cities with a lower variance of PM2.5 levels, through transfer learning. After trying this for multiple cases, it was quantitively proved that in 62.5 % of cases transfer learning improved the prediction accuracies for the later cities.

Glossary

Term	***Explanation or full form***
PM2.5	Air pollution content in particulate form that are of size 2.5 micrometer or less.
PM10	Air pollution content in particulate form that are of size between 2.5 and 10 micrometers.
ARIMA	'Auto Regressive Integrated Moving Average' is statistical regression model used on time series data for data analysis and patter recognition, as well as for forecasting upcoming values from older values (Prabhakaran, 2021).
NNs	Neural Networks
RNNs	Recurrent Neural Networks
ANNs	Artificial Neural Networks
Root Mean Square Error (RMSE)	It is the square root of the mean of the square of all of the errors, where the error is the difference between the actual target value and the predicted target value (Stephanie Glen, 2021).
Mean Absolute Error (MAE)	It is the mean of the absolute values of the individual prediction errors over all instances, where the error is the difference between the actual target value and the predicted target value (Stephanie Glen, 2016).
R2 or Coefficient of determination	It is a ratio used to show how much of a regression model's variation for a dependent variable is influenced by one or more independent variables (Jason Fernando, 2021).
Nash-Sutcliffe Efficiency index (NSE)	It is a measure that assess the amount of variance among residuals in relation to the amount of variance among the actual data, in values between 0 and 1 (AgriMetSoft, 2019).

References

Abhilash, M.S.K. *et al.* (2018) 'Time Series Analysis of Air Pollution in Bengaluru Using ARIMA Model', in G.M. Perez et al. (eds) *Ambient Communications and Computer Systems*. Singapore: Springer (Advances in Intelligent Systems and Computing), pp. 413–426. Available at: https://doi.org/10.1007/978-981-10-7386-1_36.

Acker, A. and Kreisberg, A. (2020) 'Social media data archives in an API-driven world', *Archival Science*, 20(2), pp. 105–123. Available at: https://doi.org/10.1007/s10502-019-09325-9.

AgriMetSoft (2019) *Nash Sutcliffe Model Efficiency Coefficient Calculator*. Available at: https://agrimetsoft.com/calculators/Nash%20Sutcliffe%20model%20Efficiency%20coefficient# (Accessed: 2 March 2023).

Air Quality Data in India (2015 - 2020) (no date a). Available at: https://www.kaggle.com/datasets/rohanrao/air-quality-data-in-india (Accessed: 25 September 2022).

Air Quality Data in India (2015 - 2020) (no date b). Available at: https://www.kaggle.com/rohanrao/air-quality-data-in-india (Accessed: 20 May 2022).

Ambient (outdoor) air pollution (no date). Available at: https://www.who.int/news-room/fact-sheets/detail/ambient-(outdoor)-air-quality-and-health (Accessed: 20 May 2022).

Bai, L. *et al.* (2018) 'Air Pollution Forecasts: An Overview', *International Journal of Environmental Research and Public Health*, 15(4), p. 780. Available at: https://doi.org/10.3390/ijerph15040780.

Bekkar, A. *et al.* (2021) 'Air-pollution prediction in smart city, deep learning approach', *Journal of Big Data*, 8(1), p. 161. Available at: https://doi.org/10.1186/s40537-021-00548-1.

Bertrand, H. *et al.* (2017) 'Hyperparameter Optimization of Deep Neural Networks: Combining Hyperband with Bayesian Model Selection', p. 5.

Biancofiore, F. *et al.* (2017) 'Recursive neural network model for analysis and forecast of PM10 and PM2.5', *Atmospheric Pollution Research*, 8(4), pp. 652–659. Available at: https://doi.org/10.1016/j.apr.2016.12.014.

Boznar, M., Lesjak, M. and Mlakar, P. (1993) 'A neural network-based method for short-term predictions of ambient SO2 concentrations in highly polluted industrial areas of complex terrain', *Atmospheric Environment. Part B. Urban Atmosphere*, 27(2), pp. 221–230. Available at: https://doi.org/10.1016/0957-1272(93)90007-S.

Chaloulakou, A., Assimacopoulos*, D. and Lekkas, T. (1999) 'Forecasting Daily Maximum Ozone Concentrations in the Athens Basin', *Environmental Monitoring and Assessment*, 56(1), pp. 97–112. Available at: https://doi.org/10.1023/A:1005943201063.

Chelani, A.B., Gajghate, D.G. and Hasan, M.Z. (2002) 'Prediction of Ambient PM_{10} and Toxic Metals Using Artificial Neural Networks', *Journal of the Air & Waste Management Association*, 52(7), pp. 805–810. Available at: https://doi.org/10.1080/10473289.2002.10470827.

Chen, J.X., Jiang, D.M. and Zhang, Y.N. (2019) 'A Hierarchical Bidirectional GRU Model With Attention for EEG-Based Emotion Classification', *IEEE Access*, 7, pp. 118530–118540. Available at: https://doi.org/10.1109/ACCESS.2019.2936817.

Dawson, C. (2015) *Projects in Computing and Information Systems 3rd edn: A Student's Guide*. Pearson Education.

Delavar, M. *et al.* (2019) 'A Novel Method for Improving Air Pollution Prediction Based on Machine Learning Approaches: A Case Study Applied to the Capital City of Tehran', *ISPRS International Journal of Geo-Information*, 8(2), p. 99. Available at: https://doi.org/10.3390/ijgi8020099.

Dey, S. and Chowdhury, S. (2022) 'Chapter 12 - Air quality management in India using satellite data', in R.P. Singh (ed.) *Asian Atmospheric Pollution*. Elsevier, pp. 239–254. Available at: https://doi.org/10.1016/B978-0-12-816693-2.00019-6.

Duan, Y., L.V., Y. and Wang, F.-Y. (2016) 'Travel time prediction with LSTM neural network', in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1053–1058. Available at: https://doi.org/10.1109/ITSC.2016.7795686.

Finz, G., Fronza, G. and Spirito, A. (1980) 'Multivariate Stochastic Models of Sulphur Dioxide Pollution in an Urban Area', *Journal of the Air Pollution Control Association*, 30(11), pp. 1212–1215. Available at: https://doi.org/10.1080/00022470.1980.10465170.

First Global Conference on Air Pollution and Health (2018). Available at: https://www.who.int/news-room/events/detail/2018/10/30/default-calendar/air-pollution-conference (Accessed: 8 September 2022).

Fong, I.H. *et al.* (2020) 'Predicting concentration levels of air pollutants by transfer learning and recurrent neural network', *Knowledge-Based Systems*, 192, p. 105622. Available at: https://doi.org/10.1016/j.knosys.2020.105622.

Fu, R., Zhang, Z. and Li, L. (2016) 'Using LSTM and GRU neural network methods for traffic flow prediction', in *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*. *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 324–328. Available at: https://doi.org/10.1109/YAC.2016.7804912.

Fuller, R. *et al.* (2022) 'Pollution and health: a progress update', *The Lancet Planetary Health*, 6(6), pp. e535–e547. Available at: https://doi.org/10.1016/S2542-5196(22)00090-0.

Gopu, P., Panda, R.R. and Nagwani, N.K. (2021) 'Time Series Analysis Using ARIMA Model for Air Pollution Prediction in Hyderabad City of India', in V.S. Reddy et al. (eds) *Soft Computing and Signal Processing*. Singapore: Springer (Advances in Intelligent Systems and Computing), pp. 47–56. Available at: https://doi.org/10.1007/978-981-33-6912-2_5.

Gu, Y., Li, B. and Meng, Q. (2022) 'Hybrid interpretable predictive machine learning model for air pollution prediction', *Neurocomputing*, 468, pp. 123–136. Available at: https://doi.org/10.1016/j.neucom.2021.09.051.

Gulia, S. *et al.* (2022) 'Evolution of air pollution management policies and related research in India', *Environmental Challenges*, 6, p. 100431. Available at: https://doi.org/10.1016/j.envc.2021.100431.

He, Q.-Q., Pang, P.C.-I. and Si, Y.-W. (2019) 'Transfer Learning for Financial Time Series Forecasting', in A.C. Nayak and A. Sharma (eds) *PRICAI 2019: Trends in Artificial Intelligence*. Cham: Springer International Publishing (Lecture Notes in Computer Science), pp. 24–36. Available at: https://doi.org/10.1007/978-3-030-29911-8_3.

Historical Weather Data for Indian Cities (no date a). Available at: https://www.kaggle.com/datasets/hiteshsoneji/historical-weather-data-for-indian-cities (Accessed: 25 September 2022).

Historical Weather Data for Indian Cities (no date b). Available at: https://www.kaggle.com/hiteshsoneji/historical-weather-data-for-indian-cities (Accessed: 20 May 2022).

Hooyberghs, J. *et al.* (2005) 'A neural network forecast for daily average PM10 concentrations in Belgium', *Atmospheric Environment*, 39(18), pp. 3279–3289. Available at: https://doi.org/10.1016/j.atmosenv.2005.01.050.

Huang, C.-J. and Kuo, P.-H. (2018) 'A Deep CNN-LSTM Model for Particulate Matter (PM2.5) Forecasting in Smart Cities', *Sensors*, 18(7), p. 2220. Available at: https://doi.org/10.3390/s18072220.

Jason Fernando (2021) *R-Squared Formula, Regression, and Interpretations*, *Investopedia*. Available at: https://www.investopedia.com/terms/r/r-squared.asp (Accessed: 2 March 2023).

Jeya, S. and Sankari, L. (2020) 'Air Pollution Prediction by Deep Learning Model', in *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*. *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 736–741. Available at: https://doi.org/10.1109/ICICCS48265.2020.9120932.

Kanaujia, A. *et al.* (2022) *Air Pollution in India: A Critical Assessment and Suggestive Pathways for Clean Air CSIR-NIScPR Discussion Paper Series*. Available at: https://niscpr.res.in/knowledgeproducts/discussionpapers.

Khilnani, G.C. and Tiwari, P. (2018) 'Air pollution in India and related adverse respiratory health effects: past, present, and future directions', *Current Opinion in Pulmonary Medicine*, 24(2), pp. 108–116. Available at: https://doi.org/10.1097/MCP.0000000000000463.

Kolehmainen, M., Martikainen, H. and Ruuskanen, J. (2001) 'Neural networks and periodic components used in air quality forecasting', *Atmospheric Environment*, 35(5), pp. 815–825. Available at: https://doi.org/10.1016/S1352-2310(00)00385-X.

Krause, P., Boyle, D.P. and Bäse, F. (2005) 'Comparison of different efficiency criteria for hydrological model assessment', *Advances in Geosciences*, 5, pp. 89–97.

Krishan, M. *et al.* (2019) 'Air quality modelling using long short-term memory (LSTM) over NCT-Delhi, India', *Air Quality, Atmosphere & Health*, 12(8), pp. 899–908. Available at: https://doi.org/10.1007/s11869-019-00696-7.

Kukkonen, J. *et al.* (2003) 'Extensive evaluation of neural network models for the prediction of NO2 and PM10 concentrations, compared with a deterministic modelling system and measurements in central Helsinki', *Atmospheric Environment*, 37(32), pp. 4539–4550. Available at: https://doi.org/10.1016/S1352-2310(03)00583-1.

Kumar, A. and Goyal, P. (2011) 'Forecasting of daily air quality index in Delhi', *Science of The Total Environment*, 409(24), pp. 5517–5523. Available at: https://doi.org/10.1016/j.scitotenv.2011.08.069.

Lazzeri, Francesca (2020) 'Machine Learning for Time Series Forecasting with Python®'. Available at: https://onlinelibrary.wiley.com/doi/epub/10.1002/9781119682394 (Accessed: 19 September 2022).

Leys, C. *et al.* (2013) 'Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median', *Journal of Experimental Social Psychology*, 49(4), pp. 764–766. Available at: https://doi.org/10.1016/j.jesp.2013.03.013.

Liao, K. *et al.* (2021) 'Statistical Approaches for Forecasting Primary Air Pollutants: A Review', *Atmosphere*, 12(6), p. 686. Available at: https://doi.org/10.3390/atmos12060686.

Liu, Y. *et al.* (2022) 'Air quality prediction models based on meteorological factors and real-time data of industrial waste gas', *Scientific Reports*, 12(1), p. 9253. Available at: https://doi.org/10.1038/s41598-022-13579-2.

Ma, J. *et al.* (2019) 'Improving air quality prediction accuracy at larger temporal resolutions using deep learning and transfer learning techniques', *Atmospheric Environment*, 214, p. 116885. Available at: https://doi.org/10.1016/j.atmosenv.2019.116885.

Ma, J. *et al.* (2020) 'Air quality prediction at new stations using spatially transferred bi-directional long short-term memory network', *Science of The Total Environment*, 705, p. 135771. Available at: https://doi.org/10.1016/j.scitotenv.2019.135771.

Manisalidis, I. *et al.* (2020) 'Environmental and Health Impacts of Air Pollution: A Review', *Frontiers in Public Health*, 8. Available at: https://www.frontiersin.org/articles/10.3389/fpubh.2020.00014 (Accessed: 4 September 2022).

McCollister, G.M. and Wilson, K.R. (1975) 'Linear stochastic models for forecasting daily maxima and hourly concentrations of air pollutants', *Atmospheric Environment (1967)*, 9(4), pp. 417–423. Available at: https://doi.org/10.1016/0004-6981(75)90127-4.

Most Polluted Countries 2022 (no date). Available at: https://worldpopulationreview.com/country-rankings/most-polluted-countries (Accessed: 20 May 2022).

Muthukumar, P. *et al.* (2022) 'PM2.5 Air Pollution Prediction through Deep Learning Using Multisource Meteorological, Wildfire, and Heat Data', *Atmosphere*, 13(5), p. 822. Available at: https://doi.org/10.3390/atmos13050822.

Opening address at the First WHO Global Conference on Air Pollution and Health (2018). Available at: https://www.who.int/director-general/speeches/detail/opening-address-at-the-first-who-global-conference-on-air-pollution-and-health (Accessed: 8 September 2022).

Pal, S.S. and Kar, S. (2022) 'Fuzzy transfer learning in time series forecasting for stock market prices', *Soft Computing*, 26(14), pp. 6941–6952. Available at: https://doi.org/10.1007/s00500-021-06648-7.

Prabhakaran, S. (2021) 'ARIMA Model - Complete Guide to Time Series Forecasting in Python | ML+', *Machine Learning Plus*, 22 August. Available at: https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/ (Accessed: 2 March 2023).

Public API Documentation (no date). Available at: https://www.kaggle.com/docs/api (Accessed: 25 September 2022).

Revlett, G.H. (1978) 'Ozone Forecasting Using Empirical Modeling', *Journal of the Air Pollution Control Association*, 28(4), pp. 338–343. Available at: https://doi.org/10.1080/00022470.1978.10470607.

Sekar, C. *et al.* (2016) 'Modeling and Prediction of Hourly Ambient Ozone (O3) and Oxides of Nitrogen (NOx) Concentrations Using Artificial Neural Network and Decision Tree Algorithms for an Urban Intersection in India', *Journal of Hazardous, Toxic, and Radioactive Waste*, 20(4), p. A4015001. Available at: https://doi.org/10.1061/(ASCE)HZ.2153-5515.0000270.

Stephanie Glen (2016) *Absolute Error & Mean Absolute Error (MAE), Statistics How To*. Available at: https://www.statisticshowto.com/absolute-error/ (Accessed: 2 March 2023).

Stephanie Glen (2021) *RMSE: Root Mean Square Error, Statistics How To*. Available at: https://www.statisticshowto.com/probability-and-statistics/regression-analysis/rmse-root-mean-square-error/ (Accessed: 2 March 2023).

Sun, W. and Sun, J. (2017) 'Daily PM2.5 concentration prediction based on principal component analysis and LSSVM optimized by cuckoo search algorithm', *Journal of Environmental Management*, 188, pp. 144–152. Available at: https://doi.org/10.1016/j.jenvman.2016.12.011.

Sweileh, W.M. *et al.* (2018) 'Outdoor air pollution and respiratory health: a bibliometric analysis of publications in peer-reviewed journals (1900 – 2017)', *Multidisciplinary Respiratory Medicine*, 13(1), p. 15. Available at: https://doi.org/10.1186/s40248-018-0128-5.

Tao, Q. *et al.* (2019) 'Air Pollution Forecasting Using a Deep Learning Model Based on 1D Convnets and Bidirectional GRU', *IEEE Access*, 7, pp. 76690–76698. Available at: https://doi.org/10.1109/ACCESS.2019.2921578.

Total Population by Country 2022 (no date). Available at: https://worldpopulationreview.com/countries (Accessed: 20 May 2022).

Understanding LSTM Networks -- colah's blog (no date). Available at: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (Accessed: 21 May 2022).

Victoria, A.H. and Maragatham, G. (2021) 'Automatic tuning of hyperparameters using Bayesian optimization', *Evolving Systems*, 12(1), pp. 217–223. Available at: https://doi.org/10.1007/s12530-020-09345-2.

Vohra, K. *et al.* (2021) 'Global mortality from outdoor fine particle pollution generated by fossil fuel combustion: Results from GEOS-Chem', *Environmental Research*, 195, p. 110754. Available at: https://doi.org/10.1016/j.envres.2021.110754.

WDI - The global distribution of air pollution (no date). Available at: https://datatopics.worldbank.org/world-development-indicators/stories/the-global-distribution-of-air-pollution.html (Accessed: 9 September 2022).

Weiss, K., Khoshgoftaar, T.M. and Wang, D. (2016) 'A survey of transfer learning', *Journal of Big Data*, 3(1), p. 9. Available at: https://doi.org/10.1186/s40537-016-0043-6.

Xu, X. and Yoneda, M. (2021) 'Multitask Air-Quality Prediction Based on LSTM-Autoencoder Model', *IEEE Transactions on Cybernetics*, 51(5), pp. 2577–2586. Available at: https://doi.org/10.1109/TCYB.2019.2945999.

Yang, S., Yu, X. and Zhou, Y. (2020) 'LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example', in *2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*. *2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*, pp. 98–101. Available at: https://doi.org/10.1109/IWECAI50956.2020.00027.

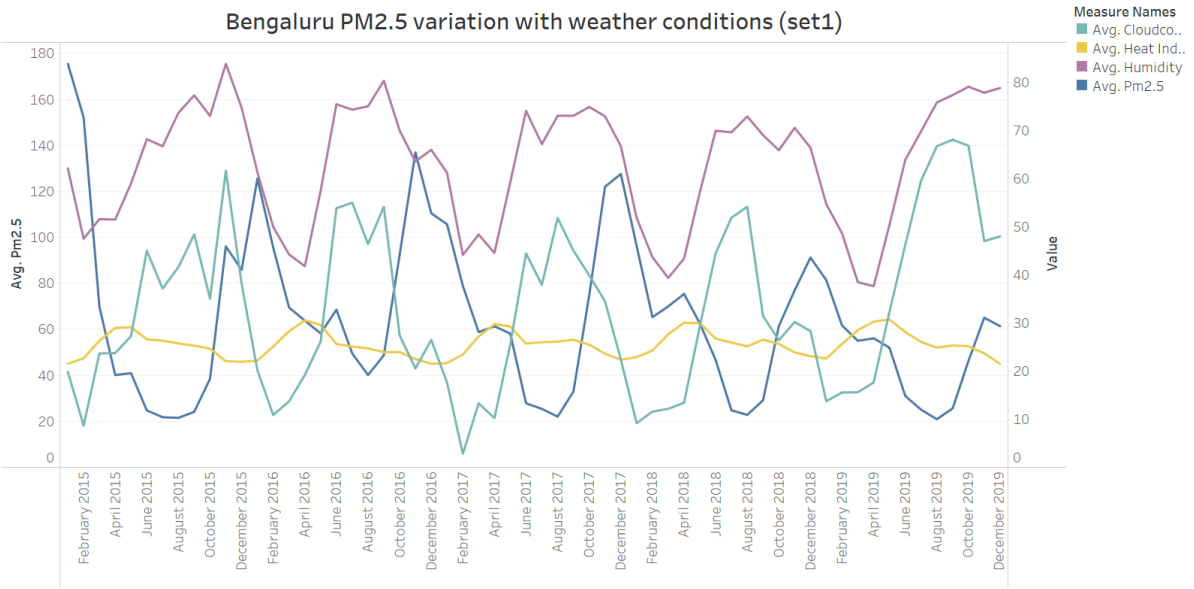
Ye, R. and Dai, Q. (2021) 'Implementing transfer learning across different datasets for time series forecasting', *Pattern Recognition*, 109, p. 107617. Available at: https://doi.org/10.1016/j.patcog.2020.107617.

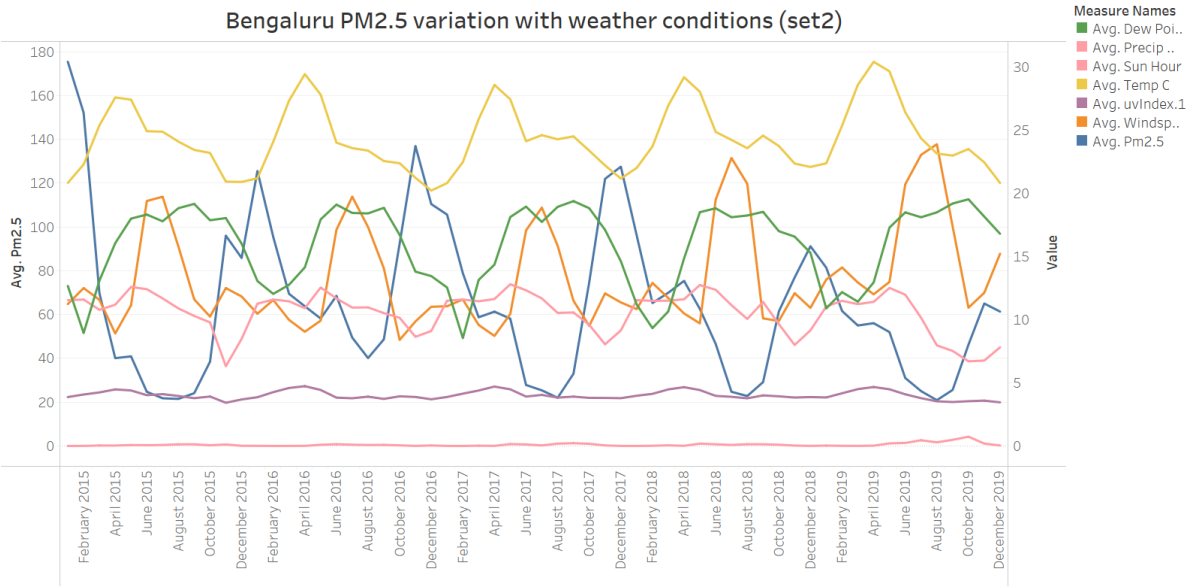
Zhai, B. and Chen, J. (2018) 'Development of a stacked ensemble model for forecasting and analyzing daily average PM2.5 concentrations in Beijing, China', *Science of The Total Environment*, 635, pp. 644–658. Available at: https://doi.org/10.1016/j.scitotenv.2018.04.040.

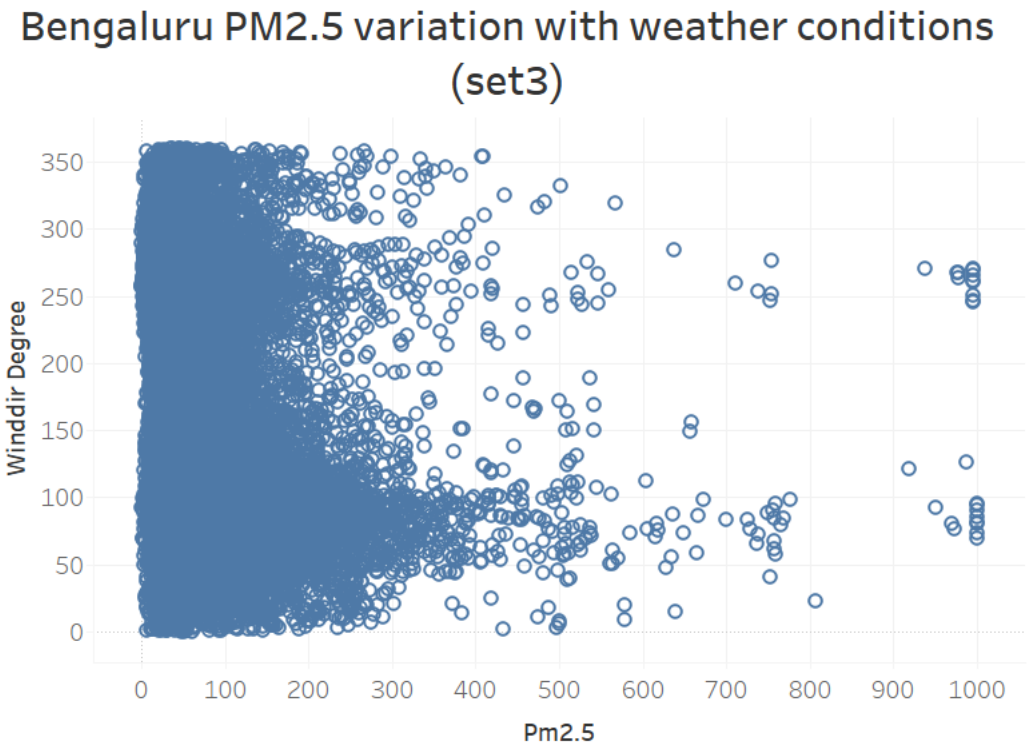
Appendix - A

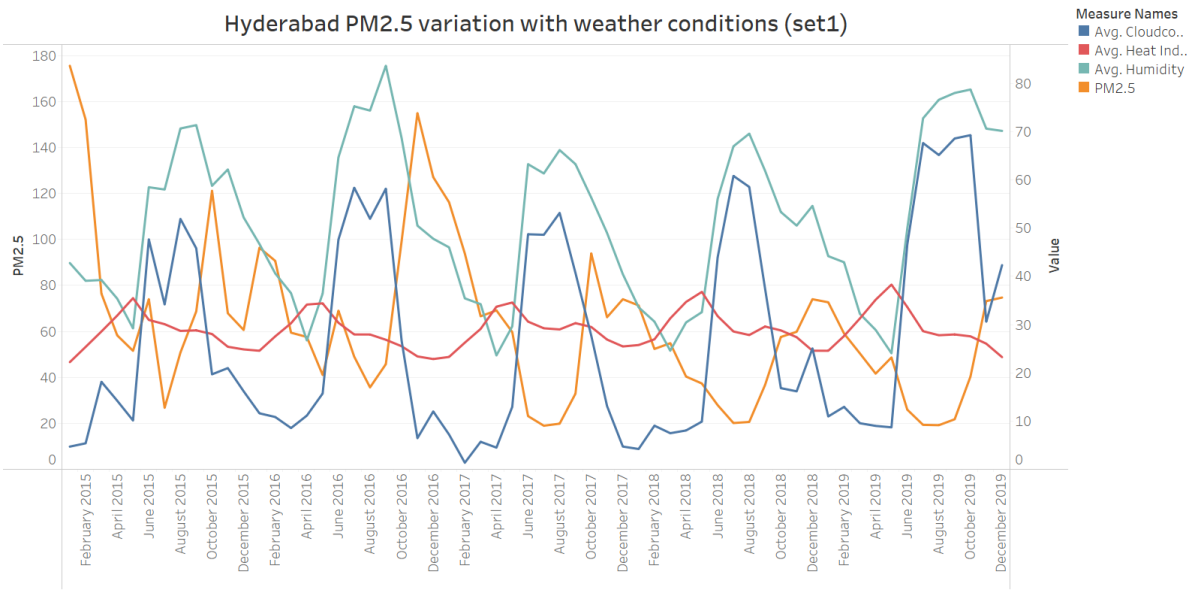
Plots of exploratory data analysis

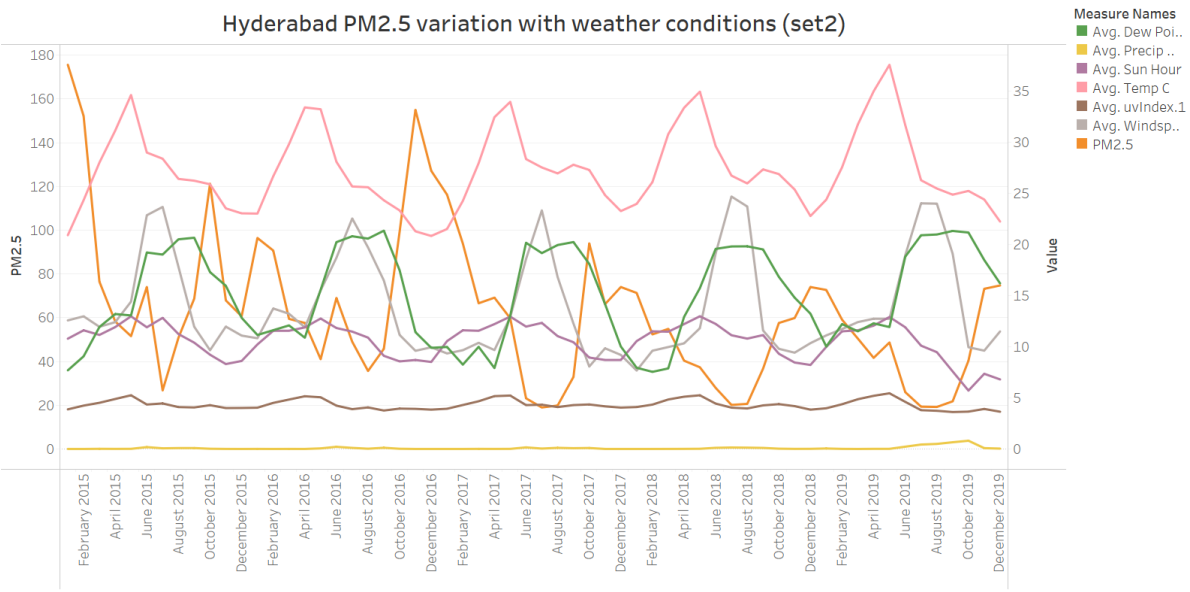
Below are the plots from exploratory data analysis of combination air pollution and meteorological data for Bengaluru, Hyderabad, Jaipur and Mumbai cities, as mention at the end of section 4.4 of chapter 4 of this report.

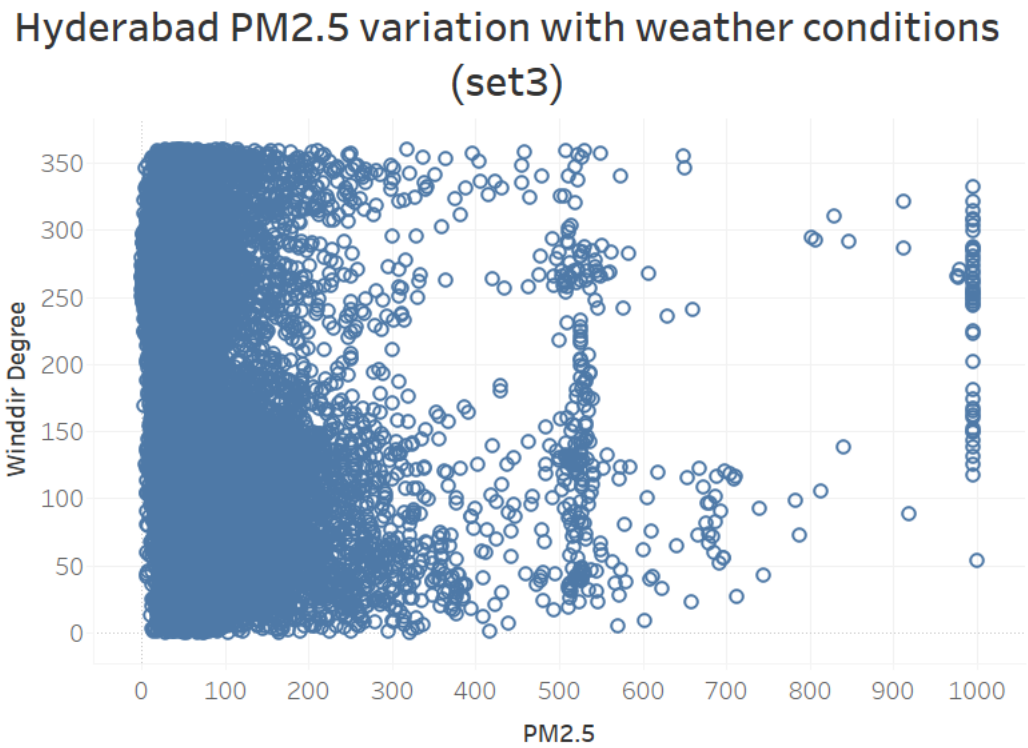


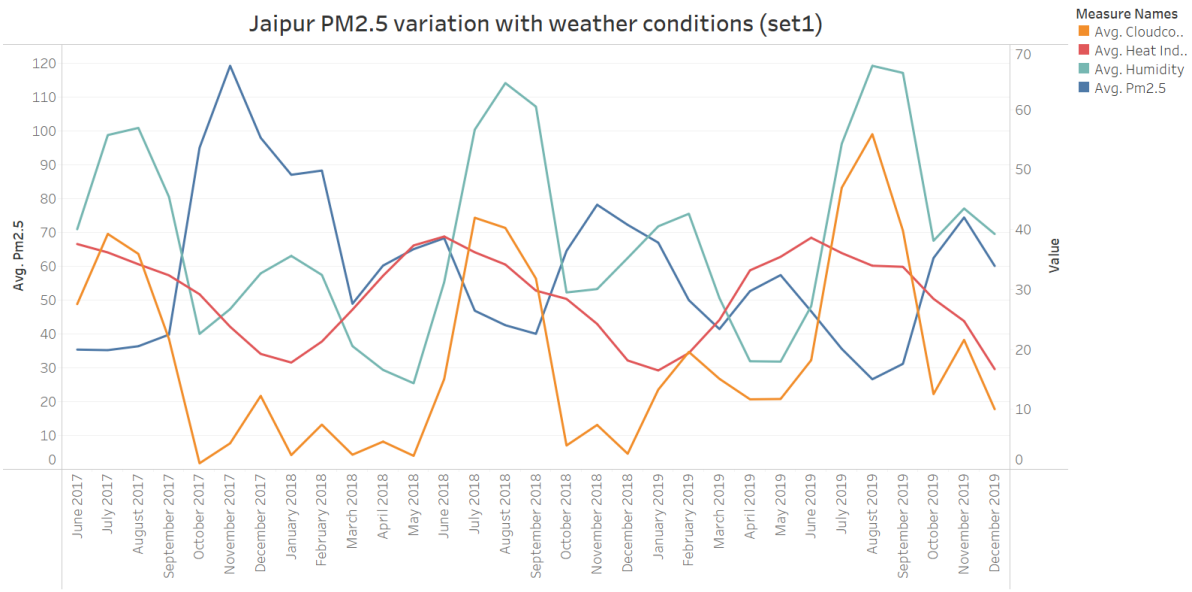


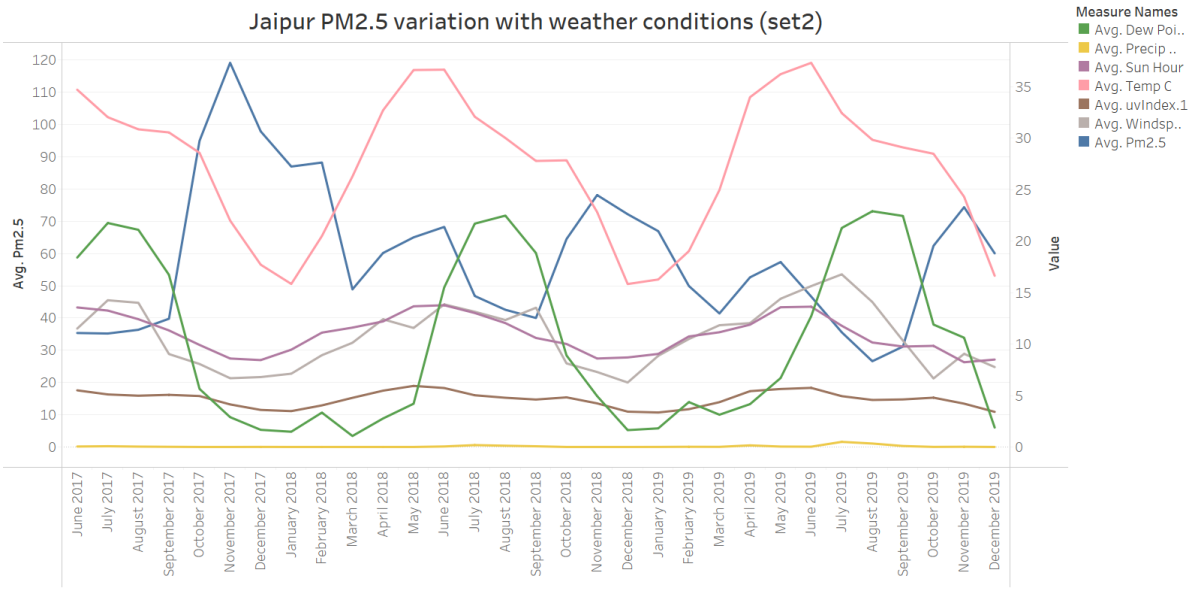

Bengaluru PM2.5 variation with weather conditions (set3)
Winddir Degree
350
300
250
200
150
100
50
0
0
100
200
300
400
500
600
700
800
900
1000
Pm2.5

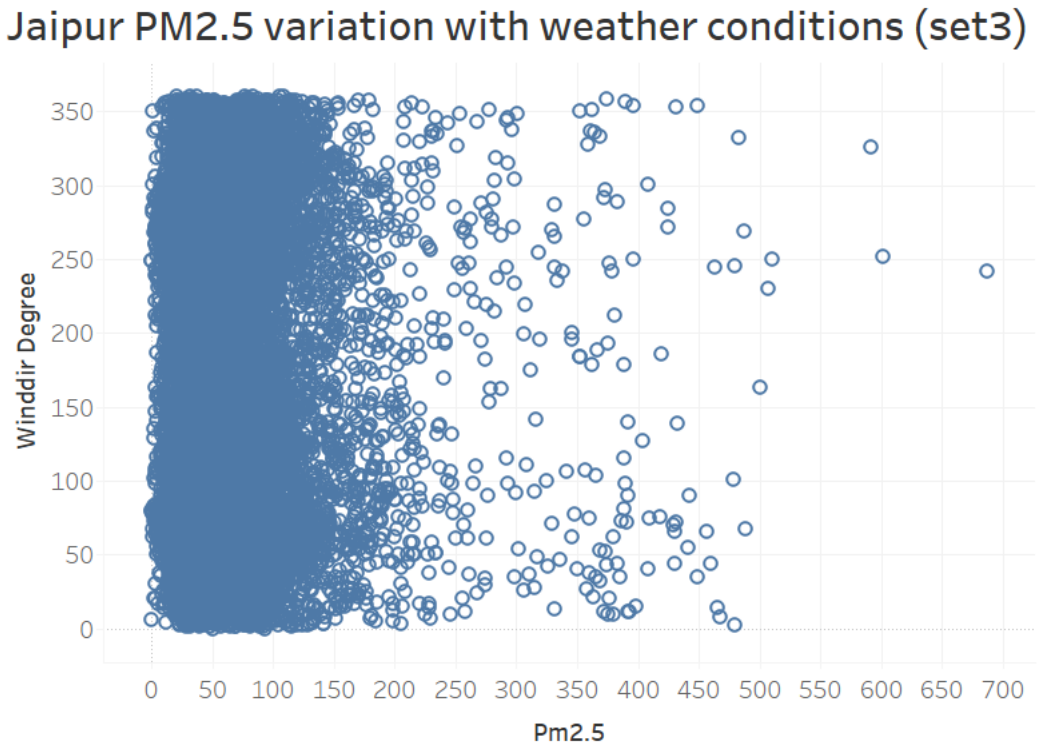

Hyderabad PM2.5 variation with weather conditions (set1)
Measure Names
Avg. Cloudco..
Avg. Heat Ind..
Avg. Humidity
PM2.5
PM2.5
Value
February 2015
April 2015
June 2015
August 2015
October 2015
December 2015
February 2016
April 2016
June 2016
August 2016
October 2016
December 2016
February 2017
April 2017
June 2017
August 2017
October 2017
December 2017
February 2018
April 2018
June 2018
August 2018
October 2018
December 2018
February 2019
April 2019
June 2019
August 2019
October 2019
December 2019

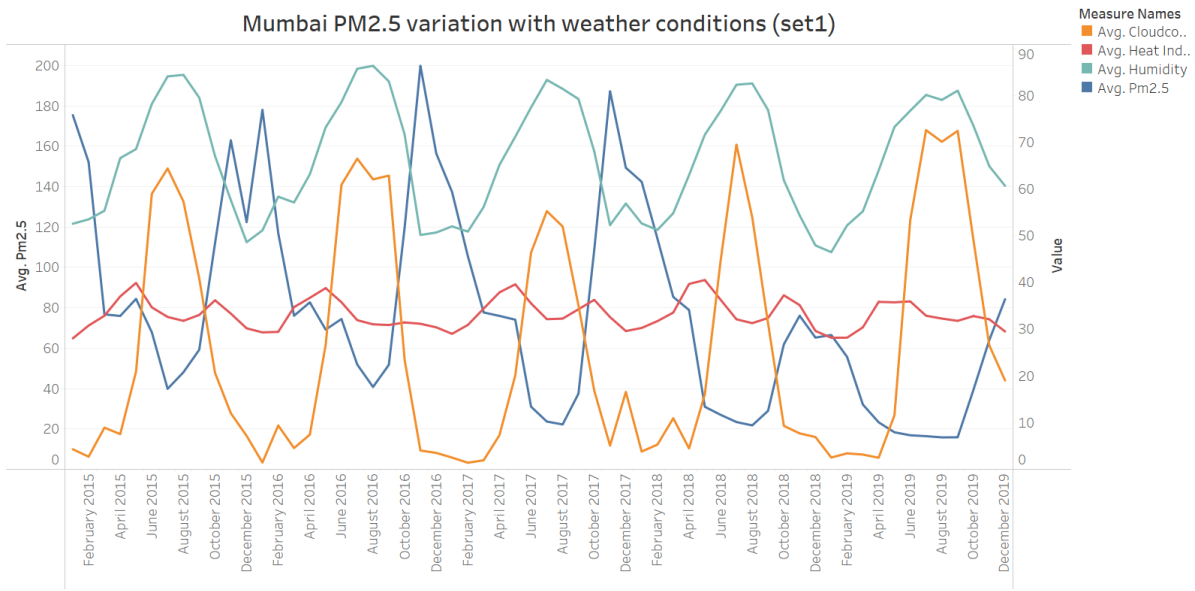

Hyderabad PM2.5 variation with weather conditions (set2)
Measure Names
Avg. Dew Poi..
Avg. Precip ..
Avg. Sun Hour
Avg. Temp C
Avg. uvIndex.1
Avg. Windsp..
PM2.5
PM2.5
Value
180
160
140
120
100
80
60
40
20
0
35
30
25
20
15
10
5
0
February 2015
April 2015
June 2015
August 2015
October 2015
December 2015
February 2016
April 2016
June 2016
August 2016
October 2016
December 2016
February 2017
April 2017
June 2017
August 2017
October 2017
December 2017
February 2018
April 2018
June 2018
August 2018
October 2018
December 2018
February 2019
April 2019
June 2019
August 2019
October 2019
December 2019

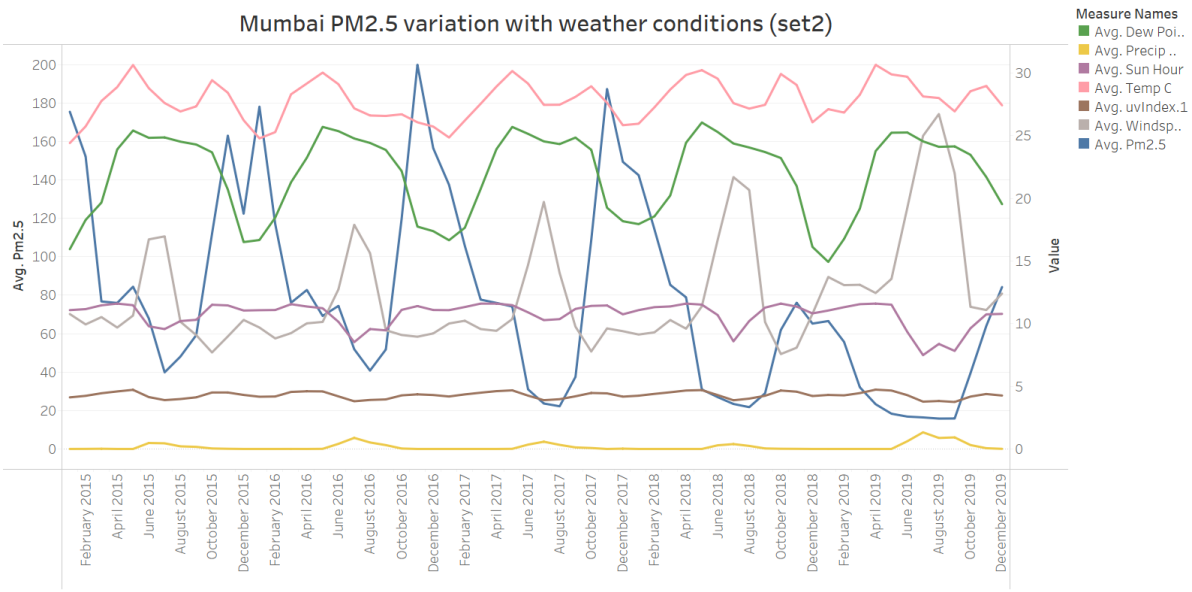

Hyderabad PM2.5 variation with weather conditions (set3)
Winddir Degree
350
300
250
200
150
100
50
0
0
100
200
300
400
500
600
700
800
900
1000
PM2.5

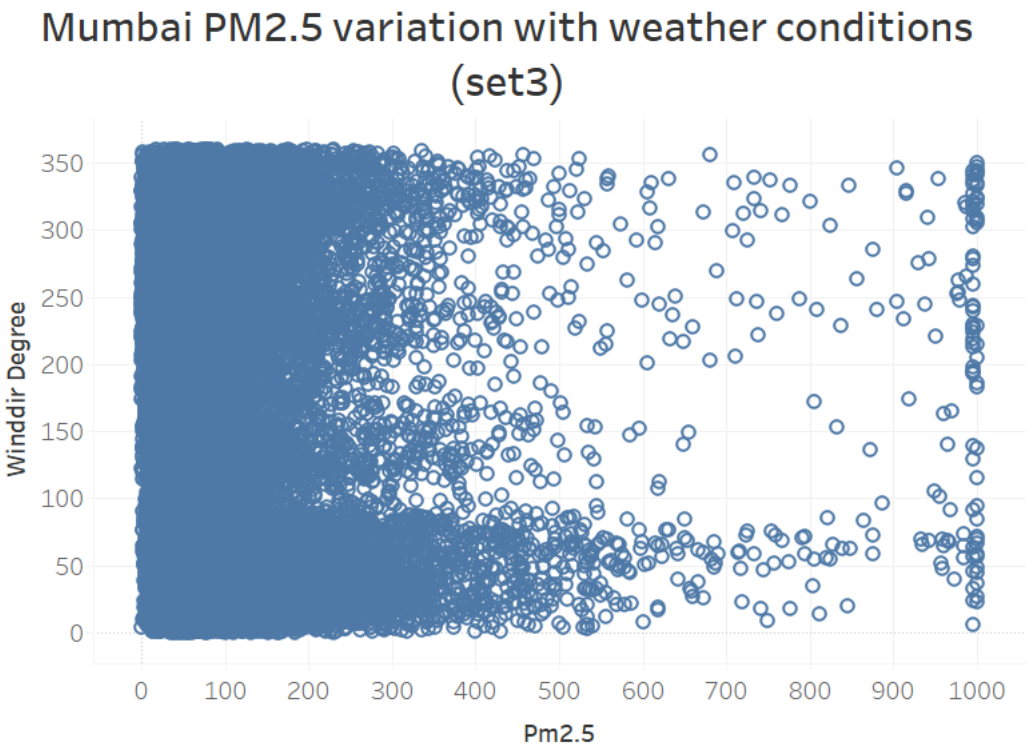

Jaipur PM2.5 variation with weather conditions (set1)
Measure Names
Avg. Cloudco..
Avg. Heat Ind..
Avg. Humidity
Avg. Pm2.5
Avg. Pm2.5
Value


Jaipur PM2.5 variation with weather conditions (set2)
Measure Names
Avg. Dew Poi..
Avg. Precip ..
Avg. Sun Hour
Avg. Temp C
Avg. uvIndex.1
Avg. Windsp..
Avg. Pm2.5
Avg. Pm2.5
Value


Jaipur PM2.5 variation with weather conditions (set3)
Winddir Degree
Pm2.5
0
50
100
150
200
250
300
350
400
450
500
550
600
650
700


Mumbai PM2.5 variation with weather conditions (set1)
Measure Names
Avg. Cloudco..
Avg. Heat Ind..
Avg. Humidity
Avg. Pm2.5
Avg. Pm2.5
Value
February 2015
April 2015
June 2015
August 2015
October 2015
December 2015
February 2016
April 2016
June 2016
August 2016
October 2016
December 2016
February 2017
April 2017
June 2017
August 2017
October 2017
December 2017
February 2018
April 2018
June 2018
August 2018
October 2018
December 2018
February 2019
April 2019
June 2019
August 2019
October 2019
December 2019


Mumbai PM2.5 variation with weather conditions (set2)
Measure Names
Avg. Dew Poi..
Avg. Precip ..
Avg. Sun Hour
Avg. Temp C
Avg. uvIndex.1
Avg. Windsp..
Avg. Pm2.5
Avg. Pm2.5
Value
200
180
160
140
120
100
80
60
40
20
0
30
25
20
15
10
5
0
February 2015
April 2015
June 2015
August 2015
October 2015
December 2015
February 2016
April 2016
June 2016
August 2016
October 2016
December 2016
February 2017
April 2017
June 2017
August 2017
October 2017
December 2017
February 2018
April 2018
June 2018
August 2018
October 2018
December 2018
February 2019
April 2019
June 2019
August 2019
October 2019
December 2019


Mumbai PM2.5 variation with weather conditions (set3)
Winddir Degree
350
300
250
200
150
100
50
0
0
100
200
300
400
500
600
700
800
900
1000
Pm2.5

Appendix – B

Code testing instructions

To run and test the coding scripts developed to perform the current project, the marker/ viewer needs to run the google Colab notebooks that are present in the google drive of the researcher. The drive folder can be accessed through the following link:

https://drive.google.com/drive/folders/1LdVEHB0wBnpZYiPt6hkVSnOeNo1uSnG3?usp=sharing

On going to the above-mentioned link, the marker/ viewer can see a 'README.txt' file and three folders viz., Code (containing colab notebooks), Data (containing intermediate data generated and a folder containing saved models), and Tableau data (containing Tableau files and image files is exploratory data analysis).

The Colab notebook files need to be run or viewed in the following order to understand the flow of the project properly:

1. Go to the "Code" folder.
2. Run the "Project EDA" notebook and close it at the end of execution.
3. Then run the "Project LSTM" notebook and close it at the end of execution.
4. Then run the "Project BiLSTM" notebook and close it at the end of execution.
5. Then run the "Project GRU" notebook and close it at the end of execution.
6. Finally, run the "Project BiGRU" notebook and close it at the end of execution.

As an additional accessibility, the researcher will attach ipynb copies of the above mentioned Colab notebooks in the additional submission area for the project.

Appendix - C

Code

This is submitted as a separate file name ‘Appendix – C - Code’ along with this report.

Appendix – D

Project proposal

Air pollution prediction over major Indian cities using neural networks

INM373 Research Methods and Professional Issues

Name: RAJANI MOHAN JANIPALLI | Student ID: 210049506 | email: Rajani.Janipalli@city.ac.uk

1. INTRODUCTION

The purpose of this project is to develop and evaluate a neural network model to predict air pollution based on past data on pollution emissions.

As per WHO (*Ambient (outdoor) air pollution*, no date), in 2019 only 1% of the global population was living in places where the ambient air pollution levels were below the limits set by it. This has led to an increase in cases of cardiovascular and respiratory health problems. As a result, 91% of the premature deaths recorded in low- and middle-income nations were due to high levels of outdoor pollution. India is one of the top five polluted countries in the world (*Most Polluted Countries 2022*, no date) and it also has the second-highest population in the world (*Total Population by Country 2022*, no date). So, the number of people affected by air pollution in India is very high. One way to save lives from severe air pollution is to predict it in advance and caution people about it so that they can take necessary precautions.

The objectives of the project are as follows:

1. Complete the literature survey of existing air pollution predictions for India and other places around the world.
2. Develop a neural network model similar to (Krishan *et al.*, 2019).
3. Evaluate and compare it with a few other types of neural network models as done by (Krishan *et al.*, 2019) and (Gu, Li and Meng, 2022).
4. Complete the project report.

The product of the work will be a neural network model that would predict air pollution with good accuracy for any similar data that is unseen by the model.

The project will have the following beneficiaries:

- Academic institutions could have this study as a basis for further research in the field of air pollution.
- Government bodies can use the results and conclusion of this work to update the measures taken to reduce the impact of air pollution.
- Web and mobile application developers can build further on the product of this study to create applications that would forecast air pollution levels.

This work will try to do all of the above by trying to answer the following research question:

Research question – Whether LSTM can be a good predictor of air pollution for spatiotemporal data with a long time range and wide spatial range?

2. CRITICAL CONTEXT

The literature review was started with the search for a recent journal paper in which a similar study was done for some regions of India. To the best of the ability of the researcher,

(Krishan *et al.*, 2019) was found to be the most recent publication in which air quality modelling was done over a small region in the capital city of India. Their paper was cited 47 times, which speaks for its literary quality. The authors of the paper belong to top universities in India. Their study developed a long short-term memory (LSTM) algorithm, which is a special type of recurring neural network (RNN), for the prediction of air quality in terms of concentrations of various pollutants, based on the data of vehicular emissions, meteorological conditions, traffic data and pollutant levels. They claimed that they were the first to model an LSTM for air quality prediction for any place in India. As per WHO (*Ambient (outdoor) air pollution*, no date), the main pollutants whose emission needs to be kept within the prescribed limits include particulate matter (PM), Ozone(O_3), Nitrogen dioxide (NO_2) and Sulphur dioxide (SO_2). (Krishan *et al.*, 2019) used emission data of all these pollutants except for SO_2, instead of which they used Carbon monoxide (CO). After developing the model, they evaluated it using different evaluation metrics which are detailed in the approaches section and then compared it with the results of (Sekar *et al.*, 2016) who developed models using artificial neural networks and decision tree algorithms to predict O_3 and NO_X concentrations for another region in India. (Krishan *et al.*, 2019) mentioned two limitations of their paper which provide a direction to pursue it in the future. The first limitation was the length of the data and the absence of data recent data as they used data for the years 2008 to 2010. The second limitation was that the data was taken only for a single station in the region, so it's only temporal but not spatiotemporal.

This project pursues to extend their study with more recent data and also for multiple cities across India, instead of a single region. Therefore, this work would try to answer the research question mentioned in the introduction section.

The literature review was then continued in search of the latest publication on air pollution prediction and a paper by (Gu, Li and Meng, 2022) was found which was cited 2 times. The authors of the paper are professors at Loughborough University, UK. Their study prepared a hybrid machine learning model by combining an artificial deep neural network with the interpretable Nonlinear Auto-Regressive Moving Average with Exogenous Input (NARMAX) model to propose a new interpretable hybrid model structure. Their study was done in Beijing, China. Unlike (Krishan *et al.*, 2019), (Gu, Li and Meng, 2022) made predictions for 1 hour ahead, 3 hours ahead and 6 hours ahead. (Gu, Li and Meng, 2022) then evaluated their model and compared its performance with LSTM from (Krishan *et al.*, 2019), ANN by likes of (Biancofiore *et al.*, 2017) and gated recurrent unit (GRU) by (Tao *et al.*, 2019). The hybrid model outperformed the rest in the 3 hours and 6 hours ahead predictions, whereas in the 1 hour ahead prediction ANN and hybrid model appeared to be equally competent. A Lasso model by (Zhai and Chen, 2018) was also considered for comparison by it performed too poor compared to the rest.

Except for (Krishan *et al.*, 2019), all the other papers mentioned above from the literature review used only meteorological data along with pollutants levels for the prediction of air pollutants concentration.

Another publication from (Xu and Yoneda, 2021, p.) was reviewed in which an LSTM autoencoder multitask learning model was prepared, that could implicitly and automatically excavate the intrinsic relevance among the pollutants in different stations. This has very advanced relevance to capturing the spatial correlation using LSTM, the second limitation of the paper by (Krishan *et al.*, 2019). Their model could jointly predict the pollutant time series at a city wide-level and could track the evolution pattern when the time series encountered drastic changes.

3. APPROACHES: METHODS & TOOLS

3.1 METHODS:

3.1.1 Data: -

As mentioned in the previous section, all the papers from the literature review worked on only meteorological data along with pollutants levels for the prediction of air pollutants concentration, except (Krishan *et al.*, 2019) who had bases from their literature review to additionally use vehicular emission data and traffic data. Although this project extends their study from the limitations of their paper, it will not use vehicular emission data and traffic data for two reasons. The first reason is to maintain consistency in the type of input data used with most of the papers from the literature review and the second reason is the availability of those two additional types of data.

This study will use the open-source data of Kaggle. The data for air pollutant levels will be collected from (*Air Quality Data in India (2015 - 2020)*, no date b) and meteorological data will be collected from (*Historical Weather Data for Indian Cities*, no date b). Data from these two sources together will provide an hourly time series from the start of 2015 to the end of 2019. So, this fulfils the first limitation of the paper by (Krishan *et al.*, 2019), by providing data for 4 years and it happens to be a recent one. From a spatial point of view, this data would cover 8 major cities of India, fulfilling the second limitation of the paper by (Krishan *et al.*, 2019).

3.1.2 Analysis: -

For analysis, it is planned to use whatever methods were applied by (Krishan *et al.*, 2019) and in case there is no method specified, methods from (Gu, Li and Meng, 2022) will be used to make the process comparable.

Since there was no mention of handling missing values by (Krishan *et al.*, 2019), missing values would be filled using interpolation methods as done by (Gu, Li and Meng, 2022). Similar is the case for handling outliers, which will be analysed and removed using smoothing and thresholding methods. Following (Krishan *et al.*, 2019) the data will be normalized to values between 0 and 1 to avoid numerical complexities. Similarly, 80% of the dataset will be used for training the models and the remaining 20% for testing the models.

3.1.3 Design: -

The product of this project will be an LSTM model that would predict air pollution for any test data that resembles training and test data used in this study in terms of data dimensions. It will be tried to create a model in a way similar to (Krishan *et al.*, 2019), with the difference in the data set used being the novelty of this work.

Since this project would consider only meteorological data along with pollutants levels, unlike (Krishan *et al.*, 2019), a single model would be created for predicting all the pollutants together as a multivariate case, with each algorithm.

To keep the process in line with (Krishan *et al.*, 2019), parameters such as learning rate, epochs and batch size would be tuned with similar values to them and also the tuning would be done through the grid search method.

Additionally for comparative analysis, it will be tried to replicate models of ANN like the one by (Biancofiore *et al.*, 2017) and GRU similar to that by (Tao *et al.*, 2019). These are

considered for comparison because they are more recent in the literature review to the best of the researcher's knowledge.

LSTM

LSTM are a special type of RNN which have a complex structure inside which lies a key component called cell state in the hidden layer (Duan, L.V. and Wang, 2016), which carries the information from history timeframe to the present time frame. This can be represented as shown in fig.1 (Duan, L.V. and Wang, 2016) below. The line at the top of the diagram, which passes straight is the cell state. The structure consists of three gates which allow only specific information through to the cell state, which is to be remembered by the cell state.

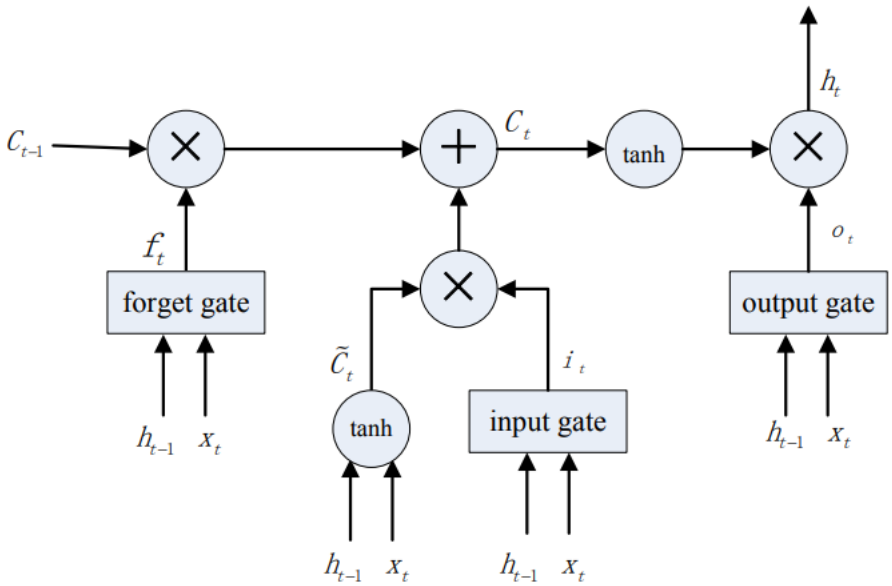


Figure 0.1: LSTM structure.

The (X) symbol indicates a multiplication operation and the (+) symbol indicates an addition operation. The information from the past output h_{t-1} and the information from present input x_t, both are fed to all three gates. All three gates consist of a sigmoid neural net (*Understanding LSTM Networks -- colah's blog*, no date), that would output how the information passes through from the gate in a value between 0 and 1. The forget gate decides which information to forget based on the difference in weights of h_{t-1} and x^t. Similarly, the input gate decides which information to add based on the difference in weights of h_{t-1} and x^t. The tanh net layer creates a candidate to update the cell state, which is then multiplied with the output of the input gate to create an update to the state. Then is added to the product of forget gate output f_t and old cell state C_{t-1}, to produce a new cell state C_t. It is then passed through a tanh net layer to produce values between -1 and 1. The output gate decides which information to output from the whole structure based on the difference in weights of h_{t-1} and x_t. Finally, the output from the LSTM structure is the product of the output of the output gate and tanh(C_t), which will then be used for future prediction as input from the future time frame will be fed to the structure.

GRU

The structure of GRU is as shown in fig 2. (Yang, Yu and Zhou, 2020), below. Structurally GRU is similar to LSTM and it is simpler to compute and implement than LSTM (Fu, Zhang and Li, 2016). It consists of a reset gate which is inside the left dashed box in the figure and

an update gate which is inside the right dashed box in the figure. The reset gate in GRU functions analogues to forget gate in LSTM. Like LSTM, the output of the present timeframe is computed using the input of the present and the output of the past timeframe in a time series. The sigmoid and tanh net layers function in a similar way as they do in an LSTM.

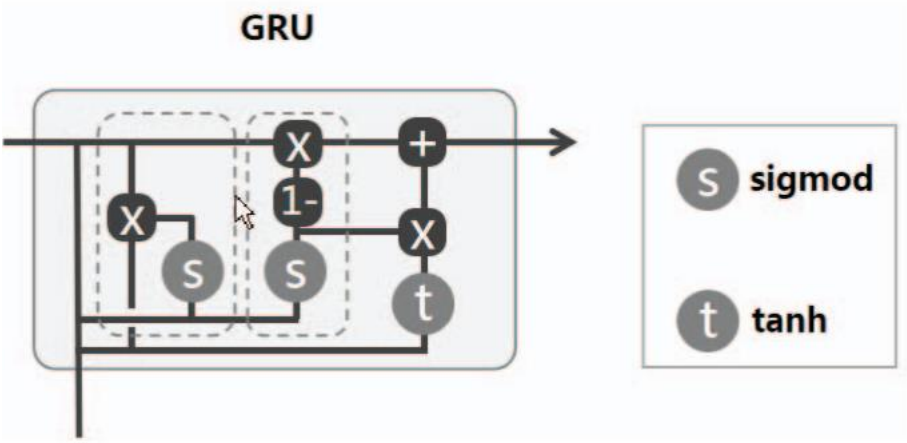


Figure 0.2: GRU structure.

The ANN will be a regular neural network with update of weight through back propagations.

3.1.4 Evaluation: -

After training the models mentioned in the design subsection above with the training data, predictions will be done for test data using the trained models.

Evaluation of the models will be done with metrics that were used by (Krishan *et al.*, 2019). These include Nash–Sutcliffe efficiency (NSE) index, correlation coefficient (CC), percentage bias (PBIAS), and root mean square error (RMSE).

NSE index is a value obtained by subtracting, the ratio of the sum of the absolute squared differences between the predicted and observed values to the variance of the observed values, from one (Krause, Boyle and Bäse, 2005).

CC is the squared ratio between the covariance and the multiplied standard deviations of the observed and predicted values (Krause, Boyle and Bäse, 2005).

PBIAS is the fraction of variance of observed values from the mean of predicted value (Krishan *et al.*, 2019).

RMSE is the square root of the fraction of difference between predicted values and the observed values (Krishan *et al.*, 2019).

Similar to (Krishan *et al.*, 2019), for LSTM the evaluation will be done for the training set first and then for the test set. This would help in identifying any overfitting and also understanding the generalized performance of the models. Then a comparative evaluation with GRU and ANN will be done for only testing data.

3.2 TOOLS:

(Krishan *et al.*, 2019) didn't mention any specification of tools used for their study. (Gu, Li and Meng, 2022) mentioned that data pre-processing was done using Python and models were developed using MATLAB. For this project, the researcher is comfortable with Python coding. So, it will be used as a tool for analysis, design and evaluation.

4. WORK PLAN

Flow diagram of work plan is as shown in fig. 3.

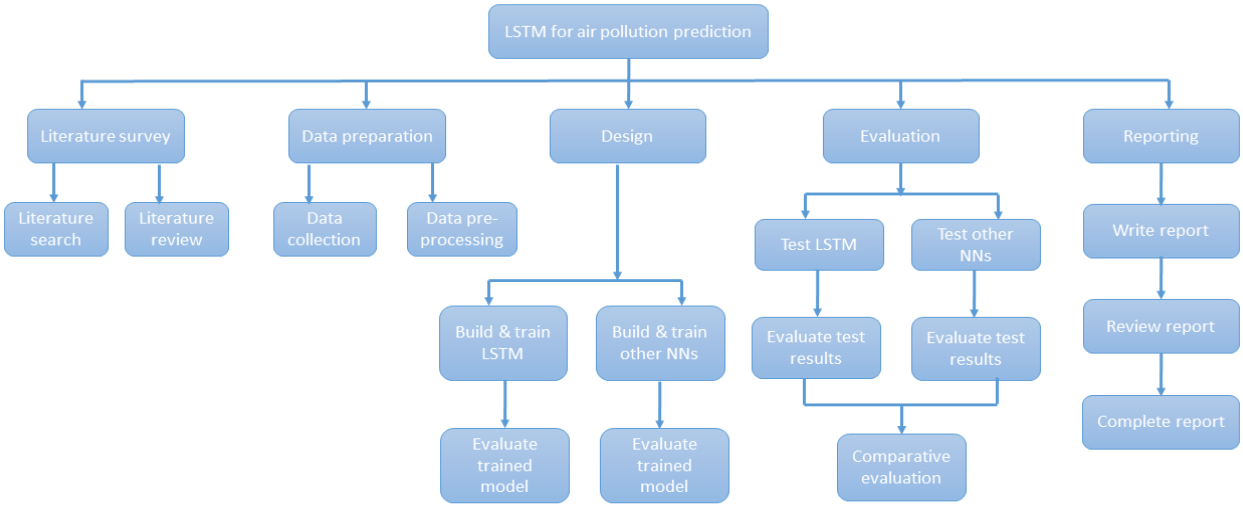


Figure 0.3: Work plan flow diagram.

Timeline of work plan is as shown in fig. 4.

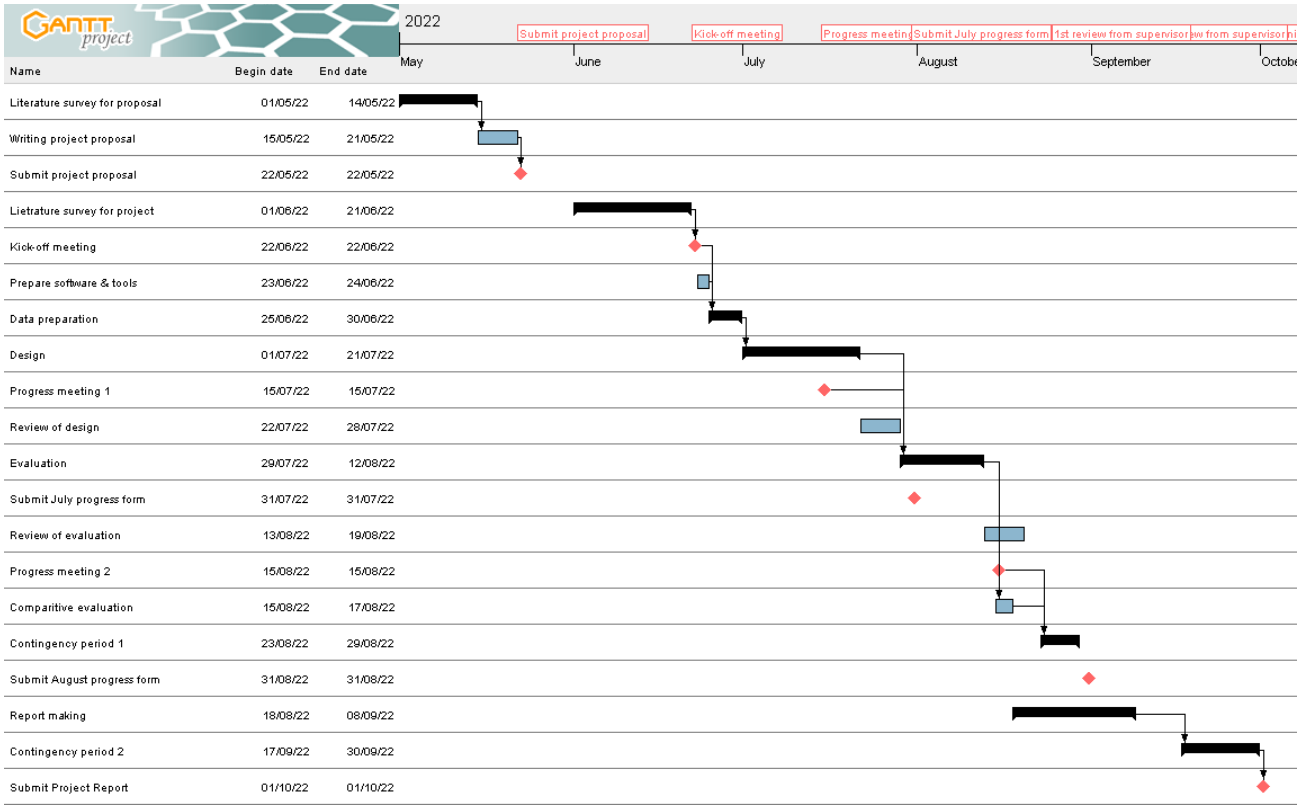


Figure 0.4: Timeline of work plan.

5. RISKS

For this project, a few technical risks have been identified. These are listed in a risk register below, along with their likelihood, consequence, impact and also how they are planned to be mitigated (Dawson, 2015).

Risk register				
Risk	**Likelihood (L) (1-3)**	**Consequence (C) (1-5)**	**Impact (L * C)**	**Mitigation strategy**
Difficulty in replicating code from reference papers	2	4	8	• Contact authors in advance. • Contact neural computing faculty. • Obtain results with code from other sources.
Computation too heavy for laptop or desktop	3	3	9	• Do modelling in MATLAB. • Obtain google Colab pro for more runtime on GPU. • Run scripts on google cloud platform with parallelization.
Not able to complete design part within stipulated time	2	3	6	• Start coding a little early. • If necessary, remove ANN or GRU from comparative study.
Results similar to reference papers cannot be achieved	3	3	9	• Perform comparative analysis with obtained results.
Any other uncertainty	2	3	6	• Contingency period 1 considered after planned completion of evaluation. • Contingency period 2 considered after planned completion of report.
Legend: Likelihood score: 1=low, 2=medium, 3=high. Consequence scores: 1=very low, 2=low, 3=medium, 4=high, 5=very high.				

References:

Abhilash, M.S.K. *et al.* (2018) 'Time Series Analysis of Air Pollution in Bengaluru Using ARIMA Model', in G.M. Perez et al. (eds) *Ambient Communications and Computer Systems*. Singapore: Springer (Advances in Intelligent Systems and Computing), pp. 413–426. Available at: https://doi.org/10.1007/978-981-10-7386-1_36.

Acker, A. and Kreisberg, A. (2020) 'Social media data archives in an API-driven world', *Archival Science*, 20(2), pp. 105–123. Available at: https://doi.org/10.1007/s10502-019-09325-9.

AgriMetSoft (2019) *Nash Sutcliffe Model Efficiency Coefficient Calculator*. Available at: https://agrimetsoft.com/calculators/Nash%20Sutcliffe%20model%20Efficiency%20coefficient# (Accessed: 2 March 2023).

Air Quality Data in India (2015 - 2020) (no date a). Available at: https://www.kaggle.com/datasets/rohanrao/air-quality-data-in-india (Accessed: 25 September 2022).

Air Quality Data in India (2015 - 2020) (no date b). Available at: https://www.kaggle.com/rohanrao/air-quality-data-in-india (Accessed: 20 May 2022).

Ambient (outdoor) air pollution (no date). Available at: https://www.who.int/news-room/fact-sheets/detail/ambient-(outdoor)-air-quality-and-health (Accessed: 20 May 2022).

Bai, L. *et al.* (2018) 'Air Pollution Forecasts: An Overview', *International Journal of Environmental Research and Public Health*, 15(4), p. 780. Available at: https://doi.org/10.3390/ijerph15040780.

Bekkar, A. *et al.* (2021) 'Air-pollution prediction in smart city, deep learning approach', *Journal of Big Data*, 8(1), p. 161. Available at: https://doi.org/10.1186/s40537-021-00548-1.

Bertrand, H. *et al.* (2017) 'Hyperparameter Optimization of Deep Neural Networks: Combining Hyperband with Bayesian Model Selection', p. 5.

Biancofiore, F. *et al.* (2017) 'Recursive neural network model for analysis and forecast of PM10 and PM2.5', *Atmospheric Pollution Research*, 8(4), pp. 652–659. Available at: https://doi.org/10.1016/j.apr.2016.12.014.

Boznar, M., Lesjak, M. and Mlakar, P. (1993) 'A neural network-based method for short-term predictions of ambient SO2 concentrations in highly polluted industrial areas of complex terrain', *Atmospheric Environment. Part B. Urban Atmosphere*, 27(2), pp. 221–230. Available at: https://doi.org/10.1016/0957-1272(93)90007-S.

Chaloulakou, A., Assimacopoulos*, D. and Lekkas, T. (1999) 'Forecasting Daily Maximum Ozone Concentrations in the Athens Basin', *Environmental Monitoring and Assessment*, 56(1), pp. 97–112. Available at: https://doi.org/10.1023/A:1005943201063.

Chelani, A.B., Gajghate, D.G. and Hasan, M.Z. (2002) 'Prediction of Ambient PM_{10} and Toxic Metals Using Artificial Neural Networks', *Journal of the Air & Waste Management Association*, 52(7), pp. 805–810. Available at: https://doi.org/10.1080/10473289.2002.10470827.

Chen, J.X., Jiang, D.M. and Zhang, Y.N. (2019) 'A Hierarchical Bidirectional GRU Model With Attention for EEG-Based Emotion Classification', *IEEE Access*, 7, pp. 118530–118540. Available at: https://doi.org/10.1109/ACCESS.2019.2936817.

Dawson, C. (2015) *Projects in Computing and Information Systems 3rd edn: A Student's Guide*. Pearson Education.

Delavar, M. *et al.* (2019) 'A Novel Method for Improving Air Pollution Prediction Based on Machine Learning Approaches: A Case Study Applied to the Capital City of Tehran', *ISPRS International Journal of Geo-Information*, 8(2), p. 99. Available at: https://doi.org/10.3390/ijgi8020099.

Dey, S. and Chowdhury, S. (2022) 'Chapter 12 - Air quality management in India using satellite data', in R.P. Singh (ed.) *Asian Atmospheric Pollution*. Elsevier, pp. 239–254. Available at: https://doi.org/10.1016/B978-0-12-816693-2.00019-6.

Duan, Y., L.V., Y. and Wang, F.-Y. (2016) 'Travel time prediction with LSTM neural network', in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1053–1058. Available at: https://doi.org/10.1109/ITSC.2016.7795686.

Finz, G., Fronza, G. and Spirito, A. (1980) 'Multivariate Stochastic Models of Sulphur Dioxide Pollution in an Urban Area', *Journal of the Air Pollution Control Association*, 30(11), pp. 1212–1215. Available at: https://doi.org/10.1080/00022470.1980.10465170.

First Global Conference on Air Pollution and Health (2018). Available at: https://www.who.int/news-room/events/detail/2018/10/30/default-calendar/air-pollution-conference (Accessed: 8 September 2022).

Fong, I.H. *et al.* (2020) 'Predicting concentration levels of air pollutants by transfer learning and recurrent neural network', *Knowledge-Based Systems*, 192, p. 105622. Available at: https://doi.org/10.1016/j.knosys.2020.105622.

Fu, R., Zhang, Z. and Li, L. (2016) 'Using LSTM and GRU neural network methods for traffic flow prediction', in *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*. *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 324–328. Available at: https://doi.org/10.1109/YAC.2016.7804912.

Fuller, R. *et al.* (2022) 'Pollution and health: a progress update', *The Lancet Planetary Health*, 6(6), pp. e535–e547. Available at: https://doi.org/10.1016/S2542-5196(22)00090-0.

Gopu, P., Panda, R.R. and Nagwani, N.K. (2021) 'Time Series Analysis Using ARIMA Model for Air Pollution Prediction in Hyderabad City of India', in V.S. Reddy et al. (eds) *Soft Computing and Signal Processing*. Singapore: Springer (Advances in Intelligent Systems and Computing), pp. 47–56. Available at: https://doi.org/10.1007/978-981-33-6912-2_5.

Gu, Y., Li, B. and Meng, Q. (2022) 'Hybrid interpretable predictive machine learning model for air pollution prediction', *Neurocomputing*, 468, pp. 123–136. Available at: https://doi.org/10.1016/j.neucom.2021.09.051.

Gulia, S. *et al.* (2022) 'Evolution of air pollution management policies and related research in India', *Environmental Challenges*, 6, p. 100431. Available at: https://doi.org/10.1016/j.envc.2021.100431.

He, Q.-Q., Pang, P.C.-I. and Si, Y.-W. (2019) 'Transfer Learning for Financial Time Series Forecasting', in A.C. Nayak and A. Sharma (eds) *PRICAI 2019: Trends in Artificial Intelligence*. Cham: Springer International Publishing (Lecture Notes in Computer Science), pp. 24–36. Available at: https://doi.org/10.1007/978-3-030-29911-8_3.

Historical Weather Data for Indian Cities (no date a). Available at: https://www.kaggle.com/datasets/hiteshsoneji/historical-weather-data-for-indian-cities (Accessed: 25 September 2022).

Historical Weather Data for Indian Cities (no date b). Available at: https://www.kaggle.com/hiteshsoneji/historical-weather-data-for-indian-cities (Accessed: 20 May 2022).

Hooyberghs, J. *et al.* (2005) 'A neural network forecast for daily average PM10 concentrations in Belgium', *Atmospheric Environment*, 39(18), pp. 3279–3289. Available at: https://doi.org/10.1016/j.atmosenv.2005.01.050.

Huang, C.-J. and Kuo, P.-H. (2018) 'A Deep CNN-LSTM Model for Particulate Matter (PM2.5) Forecasting in Smart Cities', *Sensors*, 18(7), p. 2220. Available at: https://doi.org/10.3390/s18072220.

Jason Fernando (2021) *R-Squared Formula, Regression, and Interpretations*, *Investopedia*. Available at: https://www.investopedia.com/terms/r/r-squared.asp (Accessed: 2 March 2023).

Jeya, S. and Sankari, L. (2020) 'Air Pollution Prediction by Deep Learning Model', in *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*. *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 736–741. Available at: https://doi.org/10.1109/ICICCS48265.2020.9120932.

Kanaujia, A. *et al.* (2022) *Air Pollution in India: A Critical Assessment and Suggestive Pathways for Clean Air CSIR-NIScPR Discussion Paper Series*. Available at: https://niscpr.res.in/knowledgeproducts/discussionpapers.

Khilnani, G.C. and Tiwari, P. (2018) 'Air pollution in India and related adverse respiratory health effects: past, present, and future directions', *Current Opinion in Pulmonary Medicine*, 24(2), pp. 108–116. Available at: https://doi.org/10.1097/MCP.0000000000000463.

Kolehmainen, M., Martikainen, H. and Ruuskanen, J. (2001) 'Neural networks and periodic components used in air quality forecasting', *Atmospheric Environment*, 35(5), pp. 815–825. Available at: https://doi.org/10.1016/S1352-2310(00)00385-X.

Krause, P., Boyle, D.P. and Bäse, F. (2005) 'Comparison of different efficiency criteria for hydrological model assessment', *Advances in Geosciences*, 5, pp. 89–97.

Krishan, M. *et al.* (2019) 'Air quality modelling using long short-term memory (LSTM) over NCT-Delhi, India', *Air Quality, Atmosphere & Health*, 12(8), pp. 899–908. Available at: https://doi.org/10.1007/s11869-019-00696-7.

Kukkonen, J. *et al.* (2003) 'Extensive evaluation of neural network models for the prediction of NO2 and PM10 concentrations, compared with a deterministic modelling system and measurements in central Helsinki', *Atmospheric Environment*, 37(32), pp. 4539–4550. Available at: https://doi.org/10.1016/S1352-2310(03)00583-1.

Kumar, A. and Goyal, P. (2011) 'Forecasting of daily air quality index in Delhi', *Science of The Total Environment*, 409(24), pp. 5517–5523. Available at: https://doi.org/10.1016/j.scitotenv.2011.08.069.

Lazzeri, Francesca (2020) 'Machine Learning for Time Series Forecasting with Python®'. Available at: https://onlinelibrary.wiley.com/doi/epub/10.1002/9781119682394 (Accessed: 19 September 2022).

Leys, C. *et al.* (2013) 'Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median', *Journal of Experimental Social Psychology*, 49(4), pp. 764–766. Available at: https://doi.org/10.1016/j.jesp.2013.03.013.

Liao, K. *et al.* (2021) 'Statistical Approaches for Forecasting Primary Air Pollutants: A Review', *Atmosphere*, 12(6), p. 686. Available at: https://doi.org/10.3390/atmos12060686.

Liu, Y. *et al.* (2022) 'Air quality prediction models based on meteorological factors and real-time data of industrial waste gas', *Scientific Reports*, 12(1), p. 9253. Available at: https://doi.org/10.1038/s41598-022-13579-2.

Ma, J. *et al.* (2019) 'Improving air quality prediction accuracy at larger temporal resolutions using deep learning and transfer learning techniques', *Atmospheric Environment*, 214, p. 116885. Available at: https://doi.org/10.1016/j.atmosenv.2019.116885.

Ma, J. *et al.* (2020) 'Air quality prediction at new stations using spatially transferred bi-directional long short-term memory network', *Science of The Total Environment*, 705, p. 135771. Available at: https://doi.org/10.1016/j.scitotenv.2019.135771.

Manisalidis, I. *et al.* (2020) 'Environmental and Health Impacts of Air Pollution: A Review', *Frontiers in Public Health*, 8. Available at: https://www.frontiersin.org/articles/10.3389/fpubh.2020.00014 (Accessed: 4 September 2022).

McCollister, G.M. and Wilson, K.R. (1975) 'Linear stochastic models for forecasting daily maxima and hourly concentrations of air pollutants', *Atmospheric Environment (1967)*, 9(4), pp. 417–423. Available at: https://doi.org/10.1016/0004-6981(75)90127-4.

Most Polluted Countries 2022 (no date). Available at: https://worldpopulationreview.com/country-rankings/most-polluted-countries (Accessed: 20 May 2022).

Muthukumar, P. *et al.* (2022) 'PM2.5 Air Pollution Prediction through Deep Learning Using Multisource Meteorological, Wildfire, and Heat Data', *Atmosphere*, 13(5), p. 822. Available at: https://doi.org/10.3390/atmos13050822.

Opening address at the First WHO Global Conference on Air Pollution and Health (2018). Available at: https://www.who.int/director-general/speeches/detail/opening-address-at-the-first-who-global-conference-on-air-pollution-and-health (Accessed: 8 September 2022).

Pal, S.S. and Kar, S. (2022) 'Fuzzy transfer learning in time series forecasting for stock market prices', *Soft Computing*, 26(14), pp. 6941–6952. Available at: https://doi.org/10.1007/s00500-021-06648-7.

Prabhakaran, S. (2021) 'ARIMA Model - Complete Guide to Time Series Forecasting in Python | ML+', *Machine Learning Plus*, 22 August. Available at: https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/ (Accessed: 2 March 2023).

Public API Documentation (no date). Available at: https://www.kaggle.com/docs/api (Accessed: 25 September 2022).

Revlett, G.H. (1978) 'Ozone Forecasting Using Empirical Modeling', *Journal of the Air Pollution Control Association*, 28(4), pp. 338–343. Available at: https://doi.org/10.1080/00022470.1978.10470607.

Sekar, C. *et al.* (2016) 'Modeling and Prediction of Hourly Ambient Ozone (O3) and Oxides of Nitrogen (NOx) Concentrations Using Artificial Neural Network and Decision Tree Algorithms for an Urban Intersection in India', *Journal of Hazardous, Toxic, and Radioactive Waste*, 20(4), p. A4015001. Available at: https://doi.org/10.1061/(ASCE)HZ.2153-5515.0000270.

Stephanie Glen (2016) *Absolute Error & Mean Absolute Error (MAE), Statistics How To*. Available at: https://www.statisticshowto.com/absolute-error/ (Accessed: 2 March 2023).

Stephanie Glen (2021) *RMSE: Root Mean Square Error, Statistics How To*. Available at: https://www.statisticshowto.com/probability-and-statistics/regression-analysis/rmse-root-mean-square-error/ (Accessed: 2 March 2023).

Sun, W. and Sun, J. (2017) 'Daily PM2.5 concentration prediction based on principal component analysis and LSSVM optimized by cuckoo search algorithm', *Journal of Environmental Management*, 188, pp. 144–152. Available at: https://doi.org/10.1016/j.jenvman.2016.12.011.

Sweileh, W.M. *et al.* (2018) 'Outdoor air pollution and respiratory health: a bibliometric analysis of publications in peer-reviewed journals (1900 – 2017)', *Multidisciplinary Respiratory Medicine*, 13(1), p. 15. Available at: https://doi.org/10.1186/s40248-018-0128-5.

Tao, Q. *et al.* (2019) 'Air Pollution Forecasting Using a Deep Learning Model Based on 1D Convnets and Bidirectional GRU', *IEEE Access*, 7, pp. 76690–76698. Available at: https://doi.org/10.1109/ACCESS.2019.2921578.

Total Population by Country 2022 (no date). Available at: https://worldpopulationreview.com/countries (Accessed: 20 May 2022).

Understanding LSTM Networks -- colah's blog (no date). Available at: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (Accessed: 21 May 2022).

Victoria, A.H. and Maragatham, G. (2021) 'Automatic tuning of hyperparameters using Bayesian optimization', *Evolving Systems*, 12(1), pp. 217–223. Available at: https://doi.org/10.1007/s12530-020-09345-2.

Vohra, K. *et al.* (2021) 'Global mortality from outdoor fine particle pollution generated by fossil fuel combustion: Results from GEOS-Chem', *Environmental Research*, 195, p. 110754. Available at: https://doi.org/10.1016/j.envres.2021.110754.

WDI - The global distribution of air pollution (no date). Available at: https://datatopics.worldbank.org/world-development-indicators/stories/the-global-distribution-of-air-pollution.html (Accessed: 9 September 2022).

Weiss, K., Khoshgoftaar, T.M. and Wang, D. (2016) 'A survey of transfer learning', *Journal of Big Data*, 3(1), p. 9. Available at: https://doi.org/10.1186/s40537-016-0043-6.

Xu, X. and Yoneda, M. (2021) 'Multitask Air-Quality Prediction Based on LSTM-Autoencoder Model', *IEEE Transactions on Cybernetics*, 51(5), pp. 2577–2586. Available at: https://doi.org/10.1109/TCYB.2019.2945999.

Yang, S., Yu, X. and Zhou, Y. (2020) 'LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example', in *2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*. *2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*, pp. 98–101. Available at: https://doi.org/10.1109/IWECAI50956.2020.00027.

Ye, R. and Dai, Q. (2021) 'Implementing transfer learning across different datasets for time series forecasting', *Pattern Recognition*, 109, p. 107617. Available at: https://doi.org/10.1016/j.patcog.2020.107617.

Zhai, B. and Chen, J. (2018) 'Development of a stacked ensemble model for forecasting and analyzing daily average PM2.5 concentrations in Beijing, China', *Science of The Total Environment*, 635, pp. 644–658. Available at: https://doi.org/10.1016/j.scitotenv.2018.04.040.

Research Ethics Review Form: BSc, MSc and MA Projects

Computer Science Research Ethics Committee (CSREC)
http://www.city.ac.uk/department-computer-science/research-ethics

Undergraduate and postgraduate students undertaking their final project in the Department of Computer Science are required to consider the ethics of their project work and to ensure that it complies with research ethics guidelines. In some cases, a project will need approval from an ethics committee before it can proceed. Usually, but not always, this will be because the student is involving other people ("participants") in the project.
In order to ensure that appropriate consideration is given to ethical issues, all students must complete this form and attach it to their project proposal document. There are two parts:

PART A: Ethics Checklist. All students must complete this part. The checklist identifies whether the project requires ethical approval and, if so, where to apply for approval.

PART B: Ethics Proportionate Review Form. Students who have answered "no" to all questions in A1, A2 and A3 and "yes" to question 4 in A4 in the ethics checklist must complete this part. The project supervisor has delegated authority to provide approval in such cases that are considered to involve MINIMAL risk. The approval may be ***provisional*** – *identifying the planned research as* likely to involve MINIMAL RISK. In such cases you must additionally seek ***full approval*** from the supervisor as the project progresses and details are established. ***Full approval*** must be acquired in writing, before beginning the planned research.

A.1 If you answer YES to any of the questions in this block, you must apply to an appropriate external ethics committee for approval and log this approval as an External Application through Research Ethics Online - https://ethics.city.ac.uk/		*Delete as appropriate*
1.1	Does your research require approval from the National Research Ethics Service (NRES)? *e.g. because you are recruiting current NHS patients or staff?* *If you are unsure try - https://www.hra.nhs.uk/approvals-amendments/what-approvals-do-i-need/*	**NO**
1.2	Will you recruit participants who fall under the auspices of the Mental Capacity Act? *Such research needs to be approved by an external ethics committee such as NRES or the Social Care Research Ethics Committee - http://www.scie.org.uk/research/ethics-committee/*	**NO**
1.3	Will you recruit any participants who are currently under the auspices of the Criminal Justice System, for example, but not limited to, people on remand, prisoners and those on probation? *Such research needs to be authorised by the ethics approval system of the National Offender Management Service.*	**NO**
A.2 If you answer YES to any of the questions in this block, then unless you are applying to an external ethics committee, you must apply for approval from the Senate Research Ethics Committee (SREC) through Research Ethics Online - https://ethics.city.ac.uk/		*Delete as appropriate*
2.1	Does your research involve participants who are unable to give informed consent? *For example, but not limited to, people who may have a degree of learning disability or mental health problem, that means they are unable to make an informed decision on their own behalf.*	**NO**
2.2	Is there a risk that your research might lead to disclosures from participants concerning their involvement in illegal activities?	**NO**

2.3	Is there a risk that obscene and or illegal material may need to be accessed for your research study (including online content and other material)?	**NO**
2.4	Does your project involve participants disclosing information about special category or sensitive subjects? *For example, but not limited to: racial or ethnic origin; political opinions; religious beliefs; trade union membership; physical or mental health; sexual life; criminal offences and proceedings*	**NO**
2.5	Does your research involve you travelling to another country outside of the UK, where the Foreign & Commonwealth Office has issued a travel warning that affects the area in which you will study? *Please check the latest guidance from the FCO - http://www.fco.gov.uk/en/*	**NO**
2.6	Does your research involve invasive or intrusive procedures? *These may include, but are not limited to, electrical stimulation, heat, cold or bruising.*	**NO**
2.7	Does your research involve animals?	**NO**
2.8	Does your research involve the administration of drugs, placebos or other substances to study participants?	**NO**
A.3 If you answer YES to any of the questions in this block, then unless you are applying to an external ethics committee or the SREC, you must apply for approval from the Computer Science Research Ethics Committee (CSREC) through Research Ethics Online - https://ethics.city.ac.uk/ **Depending on the level of risk associated with your application, it may be referred to the Senate Research Ethics Committee.**		*Delete as appropriate*
3.1	Does your research involve participants who are under the age of 18?	**NO**
3.2	Does your research involve adults who are vulnerable because of their social, psychological or medical circumstances (vulnerable adults)? *This includes adults with cognitive and / or learning disabilities, adults with physical disabilities and older people.*	**NO**
3.3	Are participants recruited because they are staff or students of City, University of London? *For example, students studying on a particular course or module.* *If yes, then approval is also required from the Head of Department or Programme Director.*	**NO**
3.4	Does your research involve intentional deception of participants?	**NO**
3.5	Does your research involve participants taking part without their informed consent?	**NO**
3.5	Is the risk posed to participants greater than that in normal working life?	**NO**
3.7	Is the risk posed to you, the researcher(s), greater than that in normal working life?	**NO**
A.4 If you answer YES to the following question and your answers to all other questions in sections A1, A2 and A3 are NO, then your project is deemed to be of MINIMAL RISK. **If this is the case, then you can apply for approval through your supervisor under PROPORTIONATE REVIEW. You do so by completing PART B of this form.** **If you have answered NO to all questions on this form, then your project does not require ethical approval. You should submit and retain this form as evidence of this.**		*Delete as appropriate*
4	Does your project involve human participants or their identifiable personal data? *For example, as interviewees, respondents to a survey or participants in testing.*	**NO**