SPATIAL-TEMPORAL DEEP LEARNING MODEL FOR PREDICTING PARTICULATE MATTER 2.5 (PM2.5) IN SRI LANKAN URBAN CITIES

Project Report - Group 11

Supervised By
Dr. Nalin Harischandra
Dr. G. Bowatte

Group Members
E/18/132 - Herath E.M.L.M.B.
E/18/216 - Manodya N.P.K.H.
E/18/324 - Sellahewa I.B.

Department of Electrical and Electronic Engineering

Faculty of Engineering

University of Peradeniya, Sri Lanka.

March 2024

ABSTRACT

Air pollution, particularly from fine particulate matter (PM2.5), presents a critical challenge in urban areas worldwide, including Sri Lanka. This research project endeavors to develop a spatial-temporal deep learning model specifically tailored for predicting PM2.5 concentrations in Sri Lankan urban environments. Given the severe health and environmental effects associated with PM2.5 pollution, accurate forecasting and understanding of its distribution are imperative. Our objectives encompass the development of a robust predictive model employing advanced deep learning techniques. Additionally, we aim to extend the model's applicability to regions lacking sensor networks by leveraging satellite imagery data. The overarching goal is to provide actionable insights crucial for public health interventions and urban planning initiatives. Through our research, we seek to contribute significantly to efforts aimed at enhancing public health, guiding sustainable urban development practices, and encouraging environmental protection in Sri Lanka.

This study entails a comprehensive exploration of PM2.5 concentrations across urban areas in Sri Lanka. We conduct extensive experiments involving various statistical and deep learning models to ascertain the most effective predictive approach. Our methodology involves evaluating the performance of traditional statistical models such as Autoregressive Integrated Moving Average (ARIMA) and Seasonal ARIMA (SARIMA) in conjunction with state-of-the-art deep learning architectures including Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). Additionally, we developed a hybrid model by integrating LSTM with Convolutional Neural Networks (CNN) to enhance prediction accuracy and generalizability. Through rigorous analysis and comparison, we aimed to identify the best-performing model for predicting PM2.5 concentrations.

The outcomes of our research are anticipated to yield significant benefits for Sri Lanka's urban communities and environmental management initiatives. By harnessing cutting-edge machine learning methodologies and satellite-based technologies, we aim to deliver actionable insights that empower decision-makers to enact targeted interventions aimed at mitigating PM2.5 pollution and fostering sustainable urban development. Through collaborative efforts with stakeholders, our research endeavors to drive positive change, promote public health, and safeguard the environment for current and future generations in Sri Lanka.

ACKNOWLEDGEMENTS

We extend our sincere gratitude to all those who contributed to the completion of this research project. We would like to express our heartfelt gratitude to our supervisor, Dr. Nalin Harischandra from the faculty of Engineering, University of Peradeniya, and Dr. G. Bowatte from the faculty of Allied Health Sciences, University of Peradeniya for their invaluable guidance, support, and mentorship which enabled us to complete EE 406 – Undergraduate project II on time throughout the process. We are deeply grateful to them for their encouragement, continuous guidance, and support, which inspired us to embark on this project and provided us with the necessary expertise to see it through to completion.

I would like to express my sincere gratitude to Eng. G. Attanayake and the "Building Capacity to Improve Air Quality in South Asia" project for providing the valuable dataset used in this research. Their support and contributions were instrumental in the successful completion of this project.

Additionally, we express our gratitude to Dr. Ruwan Ranaweera the course coordinator for the encouragement, and cooperation extended and especially for effectively structuring the course.

Finally, we wish to extend our sincere gratitude to the lecturers, colleagues, and all other individuals who gave the support and encouragement that empowered us to make our project a success.

Herath E.M.L.M.B. (E/18/132)
Manodya N.P.K.H. (E/18/216)
Sellahewa I.B. (E/18/324)

Department of Electrical and Electronic Engineering,
Faculty of Engineering,
University of Peradeniya.

TABLE OF CONTENTS

ABSTRACT ii
ACKNOWLEDGEMENTS iii
TABLE OF CONTENTS iv
LIST OF FIGURES vi
LIST OF TABLES viii
CHAPTER ONE 1
INTRODUCTION 1
1.1 Background Information 1
1.2 Significance of the project 2
1.3 Aim 2
1.4 Objectives 3
CHAPTER TWO 4
LITERATURE REVIEW 4
2.1 Introduction 4
2.2 Existing model 4
2.3 Validation Methods 8
CHAPTER THREE 10
METHODOLOGY 10
3.1 Introduction 10
3.2 Data collection 10
3.3 Data Preprocessing 13
3.4 Temporal Model development 18
3.5 CNN Model development 25
3.6 Hybrid Model development 34
CHAPTER FOUR 36
RESULTS 36
4.1 Introduction 36
4.2 Data Visualization 36
4.3 Autocorrelation and Partial autocorrelation in the data 37
4.4 Temporal Model Performance Evaluation 40
4.5 Temporal Model Validation 48

4.6 CNN Model Performance Evaluation 50
4.7 Hybrid Model Performance Evaluation 53
CHAPTER FIVE 55
CONCLUSION 55
REFERENCES 57

LIST OF FIGURES

Figure 1.1: PM2.5 concentrations at 1 km intervals over Sri Lanka 2
Figure 2.1: CNN-LSTM model.....5
Figure 2.2: 3D CNN-GRU model.....6
Figure 2.3: The RF–CNN joint model.....7
Figure 3.1: Overall flowchart of the methodology 10
Figure 3.2: Locations of the Stations of the Sensor Network.....11
Figure 3.3: Sample one image from each station in the dataset.....12
Figure 3.4: Number of Satellite Images Available For Each Station ……..……...12
Figure 3.5: Detected outliers using the Z-score method in Anuradhapura ...……...…13
Figure 3.6: Detected outliers using the KNN method in Anuradhapura 14
Figure 3.7: Detected outliers using the KNN method in Battaramulla ……....14
Figure 3.8: Data acquired using the KNN method in all stations ……....15
Figure 3.9: Correlation coefficients between the stations.....16
Figure 3.10: Dendrogram chart.....17
Figure 3.11: Long Short Term Memory Model Architecture.....21
Figure 3.12: GRU Model Architecture 23
Figure 3.13: Random Forest Architecture 24
Figure 3.14: Basic components of a CNN Models 26
Figure 3.15: Structure of the VGG16 Architecture 27
Figure 3.16: Structure of the ResNet50 Architecture 27
Figure 3.17: Depth-wise separable convolution layer 28
Figure 3.18: Results of the green index algorithm in Monaragala 29
Figure 3.19: Results of the green index algorithm in Kilinochchi.....30
Figure 3.20: Results of the urban index algorithm in Battaramulla.....30
Figure 3.21: Dry zone and Wet zone of Sri Lanka 31
Figure 3.22: Final CNN model architecture 33
Figure 3.23: Hybrid Model Architecture 35
Figure 4.1: Hourly Concentration of PM2.5 ($\mu g/m3$) in cluster 1 36
Figure 4.2: Hourly Concentration of PM2.5 ($\mu g/m3$) in cluster 2 37
Figure 4.3: Hourly Concentration of PM2.5 ($\mu g/m3$) in cluster 3 37
Figure 4.4: Autocorrelation of PM2.5 data in cluster 1 38
Figure 4.5: Partial autocorrelation of PM2.5 data in cluster 1.....38

Figure 4.6: Autocorrelation of PM2.5 data in cluster 2 38
Figure 4.7: Partial autocorrelation of PM2.5 data in cluster 2 39
Figure 4.8: Autocorrelation of PM2.5 data in cluster 3 39
Figure 4.9: Partial autocorrelation of PM2.5 data in cluster 3 39
Figure 4.10: Predicted values of ARIMA Model vs Test data values of cluster 1 40
Figure 4.11: Predicted values of ARIMA Model vs Test data values of cluster 2 40
Figure 4.12: Predicted values of ARIMA Model vs Test data values of cluster 3 41
Figure 4.13: Predicted values of SARIMA Model vs Test data values of cluster 1 41
Figure 4.14: Predicted values of SARIMA Model vs Test data values of cluster 2 42
Figure 4.15: Predicted values of SARIMA Model vs Test data values of cluster 3 42
Figure 4.16: Predicted values of ETS Model vs Test data values of cluster 1 43
Figure 4.17: Predicted values of ETS Model vs Test data values of cluster 2 43
Figure 4.18: Predicted values of ETS Model vs Test data values of cluster 3 44
Figure 4.19: Predicted values of LSTM Model vs Test data values of cluster 1 44
Figure 4.20: Predicted values of LSTM Model vs Test data values of cluster 2 45
Figure 4.21: Predicted values of LSTM Model vs Test data values of cluster 3 45
Figure 4.22: Predicted values of GRU Model vs Test data values of cluster 1 46
Figure 4.23: Predicted values of GRU Model vs Test data values of cluster 2 46
Figure 4.24: Predicted values of GRU Model vs Test data values of cluster 3 47
Figure 4.25: Predicted values of Random Forest Model vs Test data values of cluster 1 ... 47
Figure 4.26: Predicted values of Random Forest Model vs Test data values of cluster 2 ... 48
Figure 4.27: Predicted values of Random Forest Model vs Test data values of cluster 3 ... 48
Figure 4.28: Temporal Model Performance comparison 49
Figure 4.29: Predicted values of Basic CNN Model vs Test data values 50
Figure 4.30: Predicted values of CNN Model after adding features vs Test data values 50
Figure 4.31: Error matrix variation with each feature 51
Figure 4.32: Predicted values of CNN Model after using VGG16 vs Test data values 51
Figure 4.33: Predicted values of CNN Model after using ResNet50 vs Test data values ... 52
Figure 4.34: Predicted values of CNN Model after using MobileNet vs Test data values .. 52
Figure 4.35: Predicted values of Final CNN Model vs Test data values 53
Figure 4.36: Predicted values of Hybrid CNN-LSTM Model vs Test data values 53
Figure 4.37: Model Performance Comparison 54

LIST OF TABLES

Table 4.1: Temporal Model Validation 48
Table 4.2: Transfer Learning Model Validation 52
Table 4.3: Model Performance Comparison 54

CHAPTER ONE

INTRODUCTION

1.1 Background Information

Air pollution is a significant environmental issue with significant implications for human health and the well-being of the ecosystem worldwide. Among the various pollutants of concern, Particulate Matter 2.5 (PM2.5) stands out as a particularly harmful component due to its ability to penetrate deep into the respiratory system and cause a range of health problems, including respiratory and cardiovascular diseases. Particulate Matter 2.5 (PM2.5), a type of fine particulate matter with a diameter of 2.5 micrometers or less, is a complex mixture of solid and liquid particles suspended in the air, with sources including vehicle emissions, industrial processes, construction activities, and biomass burning. In urban areas, where population density and economic activities are high, PM2.5 pollution levels often exceed regulatory standards, posing a significant threat to public health.

Sri Lanka, like many other countries, faces the challenge of air pollution in its urban centers. Rapid urbanization, industrialization, and increased vehicular traffic have led to an increase in air pollution in cities across the country. The adverse health effects of PM2.5 pollution are well affected in these urban areas, where populations are densely concentrated.

The methodology of this study began with the collection of data from a sensor network that consists of 18 stations positioned throughout Sri Lanka. After the data collecting phase, data analysis and preprocessing were done to ensure data quality and get a basic understanding of the dataset. We were able to find some correlation among the data in between some stations. Both traditional machine learning techniques such as ARIMA and SARIMA, as well as advanced deep learning methodologies including LSTM and GRU are used to build a predictive model. After evaluating various approaches, we selected the most effective method to forecast future data trends. Furthermore, we have used satellite imagery to supplement ground-based sensor data, providing a more comprehensive understanding of spatial-temporal variations in PM2.5 concentrations. It helps to increase the estimation of PM2.5 concentrations in the areas without a sensor network and to increase the accuracy of the prediction model.

1.2 Significance of the project

This research project aims to address the need for improved prediction and understanding of PM2.5 concentrations in Sri Lankan urban cities. By using advanced techniques like spatial-temporal deep learning, along with satellite images and sensor readings, to create a system that can predict PM2.5 levels and find where pollution might be coming from. This study is significance because it could help improve public health, urban planning, and environmental protection in Sri Lanka.

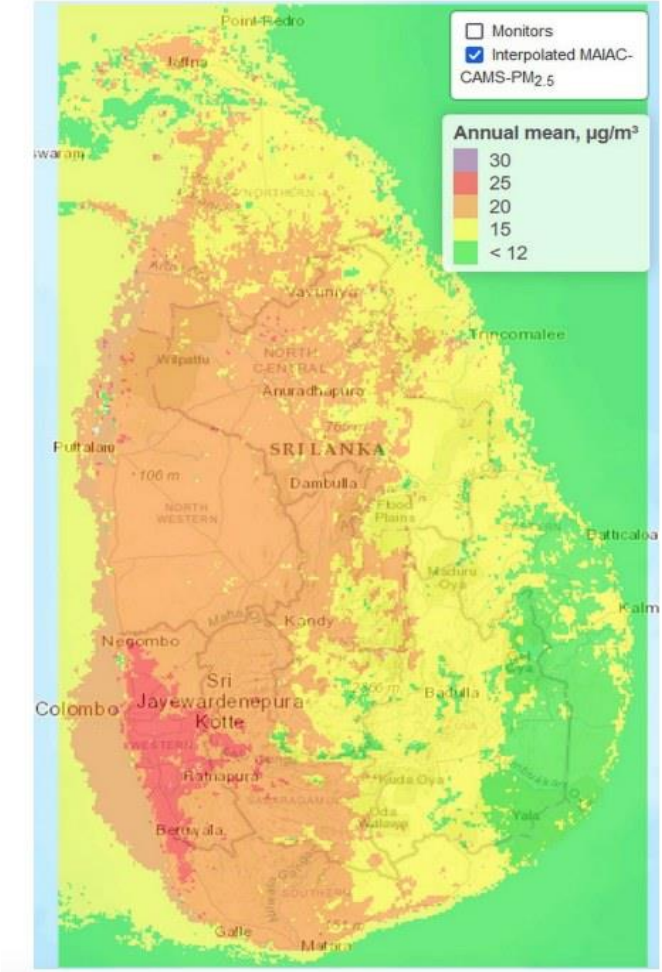


Figure 1.1: $PM_{2.5}$ concentrations at 1 km intervals over Sri Lanka.[1]

1.3 Aim

This project aims to predict and understand the distribution of Particulate Matter 2.5 (PM2.5) in Sri Lankan urban cities, providing actionable insights to protect public health, guide urban planning, and pave the way for cleaner, more sustainable urban environments.

1.4 Objectives

- To develop a powerful predictive model using deep learning techniques for accurate insights into PM2.5 concentrations in Sri Lankan urban cities, helping us identify trends, patterns, and potential sources of air pollution.
- To extend our model's predictive capabilities to areas without sensor networks using satellite imagery and advanced deep learning techniques. This will enable us to provide air quality information even in locations where traditional sensor networks are not available.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

A literature review allows us to gain a comprehensive understanding of the current state of knowledge in the field, encompassing previous studies, methodologies, and findings related to PM2.5 prediction and spatial-temporal modeling. Furthermore, reviewing literature allows us to identify best practices, challenges, and lessons learned from previous studies and analytical approaches that are informed by the experiences of others.

2.2 Existing models

2.2.1 Introduction

Recently, deep learning has become increasingly widely used in the fields of artificial intelligence and big data. The reason is that it can learn effective feature representations from a large amount of input data and excavate the profound features of the data. Hence, the study of urban air quality concentration prediction based on deep learning has been popularized in research. At present, the PM2.5 prediction model is primarily divided into the classic model of time series estimation and the combined model of the spatiotemporal forecast. Existing time-series models, like the auto-regressive integrated moving average model (ARIMA), seasonal ARIMA can capture the temporal dependencies very well. Recurrent neural networks (RNNs) have been used successfully for sequence learning tasks. The incorporation of long short-term memory (LSTM), or gated recurrent unit (GRU), enables RNNs to learn long-term temporal dependency. For capturing the spatial variations of the PM2.5 concentration models such as CNNs are widely used. CNNs can be combined with other types of RNN architectures, to create hybrid models that capture both spatial and temporal dependencies in the PM2.5 data.

2.2.2 LSTM-CNN models

LSTM networks are a specialized type of recurrent neural network (RNN) architecture that excels at modeling sequential data, making them well-suited for time series forecasting tasks such as predicting PM2.5 concentrations.

By leveraging the memory capabilities of LSTM units, the model can effectively capture complex temporal patterns and dependencies in the PM2.5 data. This capability is crucial in dealing with the dynamic nature of air pollution data, which often exhibits significant fluctuations over time. According to the study, the most common and widely used model for capturing the spatial and temporal variations in the PM2.5 data is the LSTM-CNN model, which has proven to be particularly effective in handling high-dimensional data.

The LSTM-CNN model is used most of the time with different improvements to the LSTM-CNN model. Some models have used batch normalization, Scaled Exponential Linear Units (SELU), and early stopping techniques to improve the training efficiency and accuracy.[1]

Another research used mutual information (MI) estimator to select the spatiotemporal feature vector (STFV) that reflects both linear and nonlinear correlations between the target PM2.5 concentration. These improvements to the model have improved the stability and prediction performance of the model, ensuring that it can accurately reflect the underlying relationships in the data. The integration of MI estimation helps in fine-tuning the feature selection process, leading to a more precise and effective prediction system.[2]

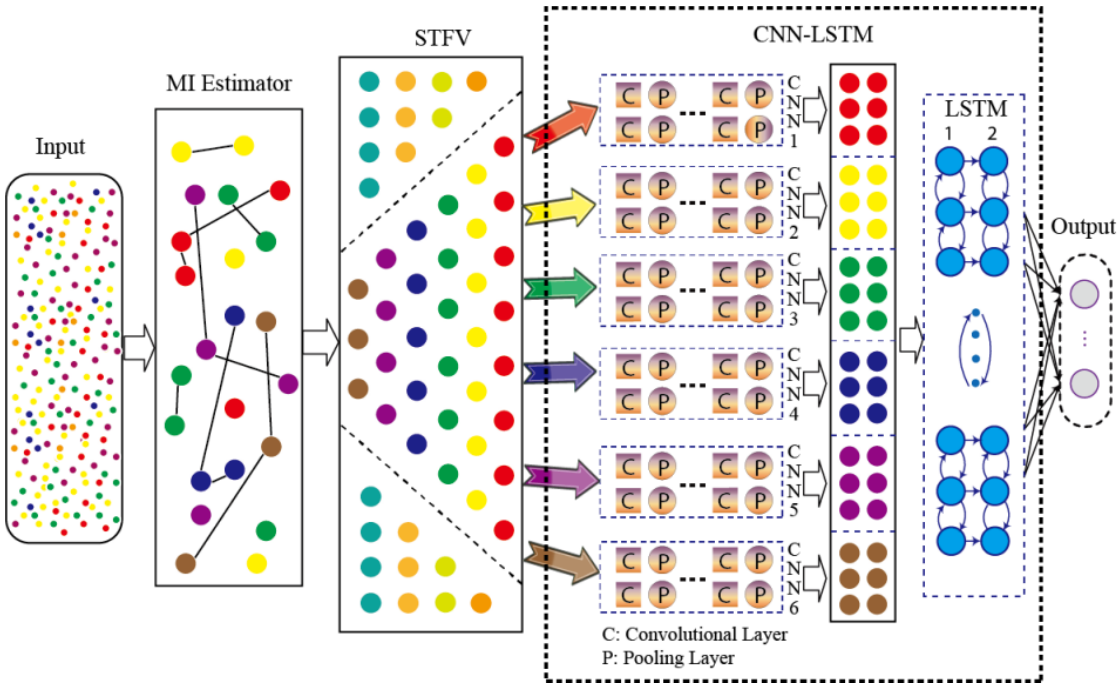


Figure 2.1: CNN-LSTM model [2]

2.2.3 GRU-CNN model

GRUs are a type of recurrent neural network (RNN) architecture, similar to Long Short-Term Memory (LSTM) units, designed to address the vanishing gradient problem commonly encountered in traditional RNNs. GRUs are particularly well-suited for modeling sequential data with long-range dependencies, making them ideal for tasks such as time series forecasting. Additionally, GRUs have a simpler structure compared to LSTMs, which often leads to faster training times while maintaining comparable performance. The combined GRU-CNN model is also a common method used in the spatial-temporal PM2.5 forecasting models, effectively capturing both spatial and temporal dependencies.

A PM2.5 prediction research based on the GRU-CNN model used air pollution and meteorological data from several stations in Tehran, Iran, and learned the spatial and temporal patterns of PM2.5 using 3D CNN and GRU layers. The integration of 3D CNN allows for a more comprehensive analysis of spatial data by considering neighboring regions simultaneously. The model also uses dynamic time warping (DTW) to find the most similar stations and process them together, enhancing its ability to generalize across different locations. Additionally, this model has achieved good results compared to LSTM-CNN models, demonstrating the effectiveness of GRUs in specific contexts where speed and simplicity are crucial factors. [3]

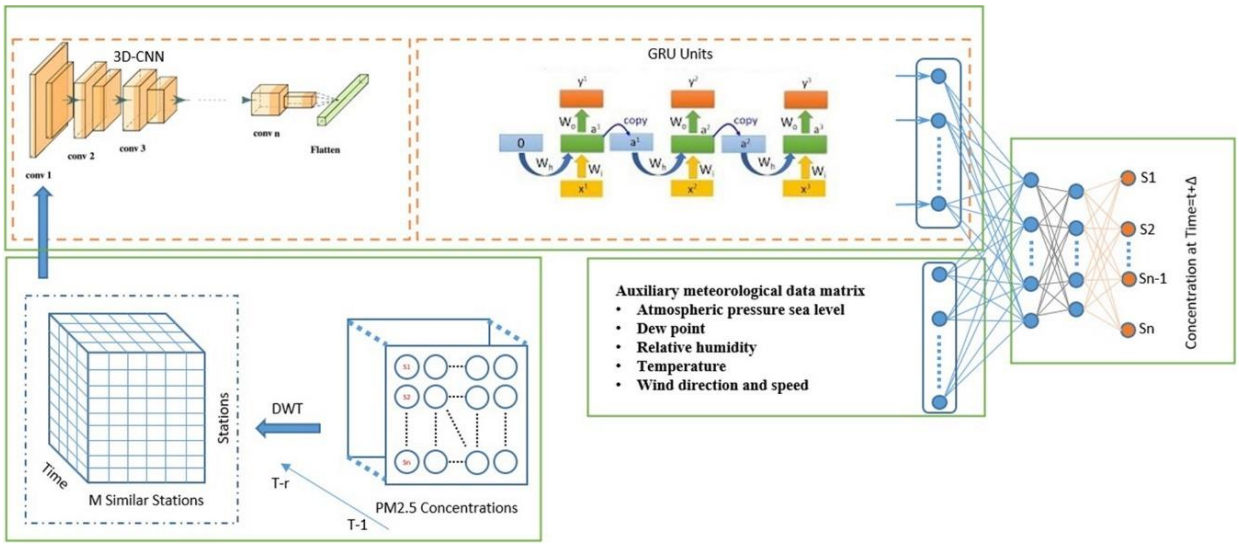


Figure 2.2: 3D CNN-GRU model [3]

2.2.4 RF-CNN model

Random Forest is a strong ensemble learning algorithm, well-known for its adaptability and durability when working with complex datasets. Its ability to handle large amounts of data with higher dimensionality makes it an attractive option for spatial-temporal PM2.5 forecasting models. The RF combined with CNN can be particularly powerful, leveraging the strengths of both machine learning and deep learning techniques for more accurate predictions.

Some researchers have tried the RF-CNN models for forecasting PM2.5 concentrations, with promising results. The RF part uses temperature, relative humidity, and sea-level pressure to predict a PM2.5 baseline map, which serves as a coarse estimate of air quality across a large area. Meanwhile, the CNN part uses high-resolution satellite images to predict the PM2.5 residuals at a 300 m resolution, capturing finer spatial details that the RF model might miss. The final PM2.5 map is obtained by adding the RF-predicted baseline and the CNN-predicted residual, resulting in a detailed and accurate representation of air quality across different regions. [4]

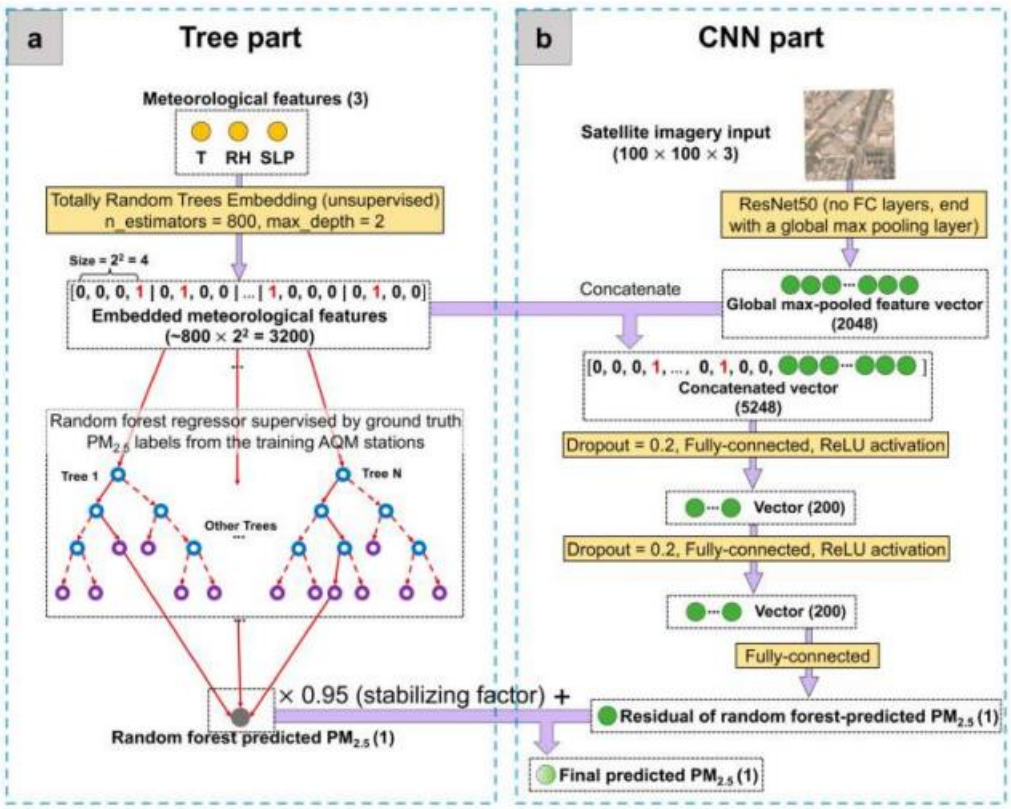


Figure 2.3: The RF–CNN joint model [4]

2.3 Validation Methods

2.3.1 Mean Absolute Error (MAE)

Mean Absolute Error (MAE) is a metric used to evaluate the accuracy of a predictive model. It measures the average absolute difference between the predicted values and the actual values. The formula for Mean Absolute Error is:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \acute{y})$$

where *n* is the number of observations or data points, y_i is the actual or observed value for the i-th data point and $\acute{y}$ is the predicted value for the i-th data point.

MAE provides a simple and intuitive measure of how well a model is performing in terms of absolute errors. It is expressed in the same units as the original data, making it easy to interpret. A lower MAE indicates better accuracy, as it means the model's predictions are closer to the actual values on average.

2.3.2 Root Mean Square Error (RMSE)

Root Mean Squared Error (RMSE) is another metric used to evaluate the accuracy of a predictive model. It is similar to Mean Absolute Error (MAE) but gives more weight to larger errors by taking the square root of the average of squared differences between the predicted values and the actual values. The formula for RMSE is:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \acute{y})^2}$$

Where *n* is the number of observations or data points, y_i is the actual or observed value for the i-th data point and $\acute{y}$ is the predicted value for the i-th data point.

The main difference between MAE and RMSE lies in the squaring of the errors in RMSE. Squaring the errors penalizes larger errors more heavily than smaller errors. RMSE is useful when you want to give more importance to outliers or significant deviations between predicted and actual values.

2.3.2 R^2 Value (Coefficient of Determination)

A statistic used to assess the effectiveness of regression models, particularly deep learning models that carry out regression tasks, is the R-squared (R^2) value, sometimes referred to as the coefficient of determination. It indicates how well a model's expected outputs match the actual data.

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

y_i = actual values

$\hat{y}_i$= predicted values

$\bar{y}$ = mean of the actual values

Interpretation:

$R^2 = 1$: Perfect fit. The model predictions match the actual data exactly.

$R^2 = 0$: The model's predictions are no better than simply predicting the mean of the data.

$R^2 < 0$: The model performs worse than predicting the mean, indicating poor model fit.

CHAPTER THREE

METHODOLOGY

3.1 Introduction

This chapter describes the main steps of the methodology of this research and the overall flowchart of the methodology is given in Figure 3.1.

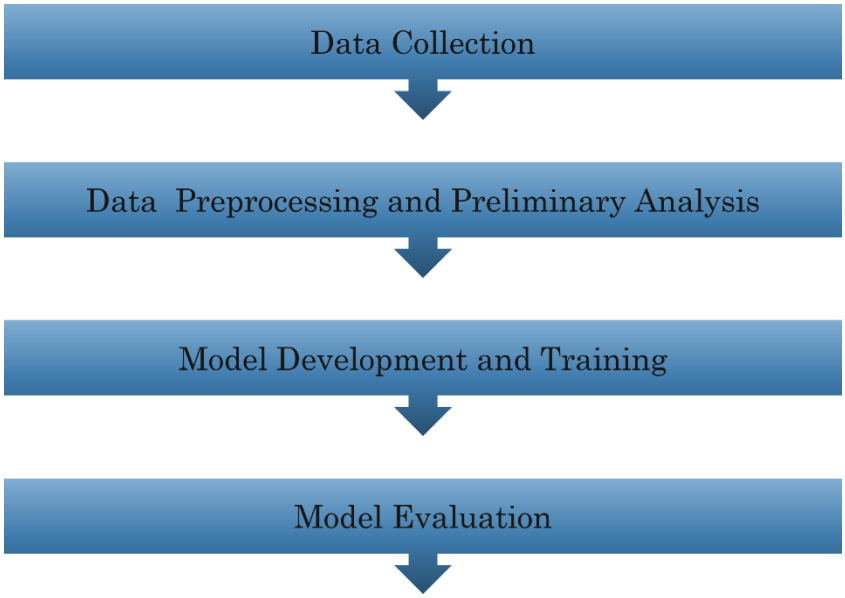


Figure 3.1: Overall flowchart of the methodology

3.2 Data collection

3.2.1 Temporal Data collection

Throughout Sri Lanka, a network of eighteen air quality monitoring sites covering important rural and urban areas is used to gather temporal data for this project. These stations are located in cities including Jaffna, Mannar, Kandy, Kilinochchi, Kanthale, Polonnaruwa, Anuradhapura, Mullaitivu, Katubedda, Galle, Hambantota, Matara, Jaffna, and Point Pedro. Parts per million (ppm) of PM2.5 are measured by BlueSky Air Quality Monitors, which are installed in every station. Hourly records of meteorological information, including temperature and relative humidity, are made in addition to PM2.5 readings. These parameters are essential to comprehending how environmental influences affect PM2.5 levels over time.

We collected hourly data for around six months at each location, thus each station has about 4,320 data points (24 hours × 30 days × 6 months). Nevertheless, the dataset has certain gaps because of problems like power outages and WiFi connectivity problems at the monitoring sites. These disruptions made it difficult to collect data continuously, which resulted in gaps in some time periods. Despite this, the temporal data provides a strong basis for the time series analysis employed in the research and provides insightful information about the time-dependent behaviour of PM2.5 concentrations and environmental circumstances.

Figure 3.2: Locations of the Stations of the Sensor Network

3.2.2 Spatial Data collection

The project integrates not only time data but also spatial data obtained from satellite imaging, which is focused on the same 18 stations that are utilized for measurements conducted on the ground. Time stamps, dates, station numbers, satellite indices, PM2.5 readings, and satellite images are all included in this spatial data. A total of 1,422 images from satellites were gathered, with each photo covering a 1 km by 1 km region surrounding the monitoring stations. Every day at the same time, these images were taken, giving a regular glimpse of the surroundings. The images were then resized to 224 x 224 pixels with three colour channels, which was the conventional size for resizing images for use as input in the CNN model for this research.

To guarantee data quality, pictures containing clouds or other visual impediments were removed during preprocessing. This process was required to eliminate extraneous noise from the dataset so that the model could concentrate on pertinent spatial variables like vegetation, land cover, and urban growth. Notwithstanding the difficulties posed by obscured or missing images, the spatial data gathered offers vital insights into the geographic variables affecting PM2.5 concentrations in various parts of Sri Lanka. By adding to the temporal observations, this data makes it possible to analyze air quality more thoroughly.

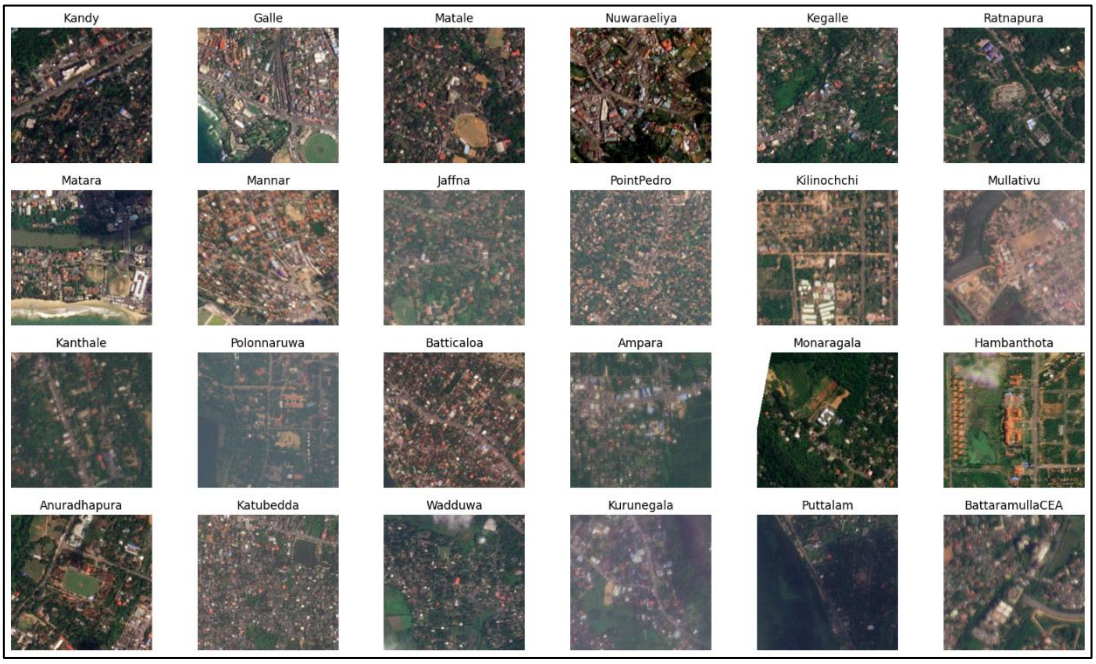


Figure 3.3: Sample one image from each station in the dataset

Station	No:of Data
Kandy	145
Galle	90
Matara	88
Mannar	20
Jaffna	53
PointPedro	62
Kilinochchi	56
Mullativu	58
Kanthale	86
Polonnaruwa	99
Batticaloa	69
Ampara	97
Monaragala	101
Hambanthota	53
Anuradhapura	91
Katubedda	9
Wadduwa	174
BattaramullaCEA	71

Figure 3.4: Number of Satellite Images Available for Each Station.

3.3 Data Preprocessing and Preliminary Analysis

3.3.1 Introduction

After collecting data, it is essential to perform preprocessing and preliminary analysis on the collected data to ensure that the data is clean, consistent, and appropriately formatted, laying a solid foundation for accurate modeling and analysis. By detecting and handling missing values, and outliers, preprocessing mitigates the risk of biased or misleading results, thereby enhancing the reliability and validity of the data. preliminary analysis, including correlation analysis, clustering, and seasonal, trend pattern identification, offers valuable insights into the spatial and temporal dynamics of PM2.5 pollution. Also, preprocessing s allows us to normalize the data, ensuring that features are on a comparable scale and optimizing the performance of the model.

3.3.2 Preprocessing

3.3.2.1 Outlier detection and treatment

There are different techniques to remove outliers. As the first step, the data points which deviate significantly from the majority of the data can be removed from the visual inspection. In advance, statistical methods such as the z-score method and inter-quartile range (IQR) method can be used to remove outliers. Also, there are machine learning algorithms like K-Nearest neighbor and isolation forest.

The Z-score method is a statistical approach employed for detecting and eliminating outliers within a dataset. This technique involves calculating the mean and standard deviation of the dataset, and then determining the Z-score for each data point by measuring its deviation from the mean in terms of standard deviations.

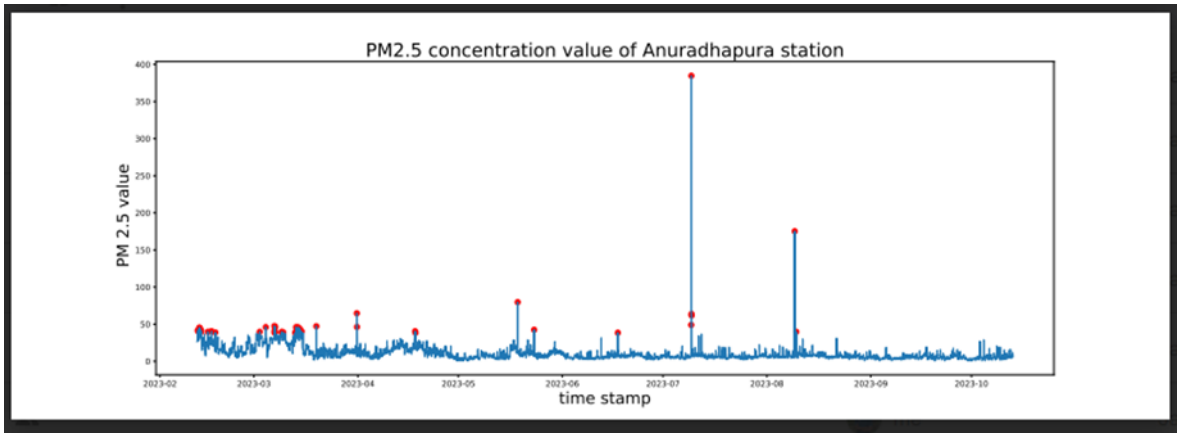


Figure 3.5: Detected outliers using the Z-score method in Anuradhapura

By setting a predefined threshold, often at Z-scores beyond 3 or -3, data points that fall significantly outside the expected range are identified as outliers. This method was not suitable for the PM2.5 data set. Because most of the peak events were considered as the outliers. In the following figure, marked red points are the outliers identified using the Z-score method.

Then K-Nearest neighbor method was considered. In this method, it identifies the outliers by measuring the distance between each data point and its K nearest neighbors. By setting a predefined threshold, if the distance is greater than the defined threshold data point is treated as an outlier. This method showed comparatively better results, it only detected very high erroneous values as outliers. In the following figure, marked red points are the outliers identified using the KNN method.

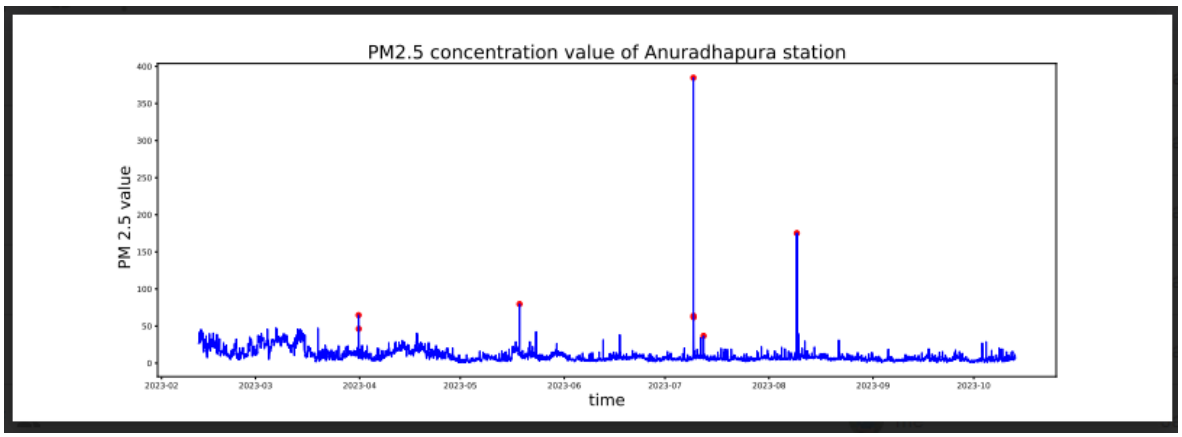


Figure 3.6: Detected outliers using the KNN method in Anuradhapura

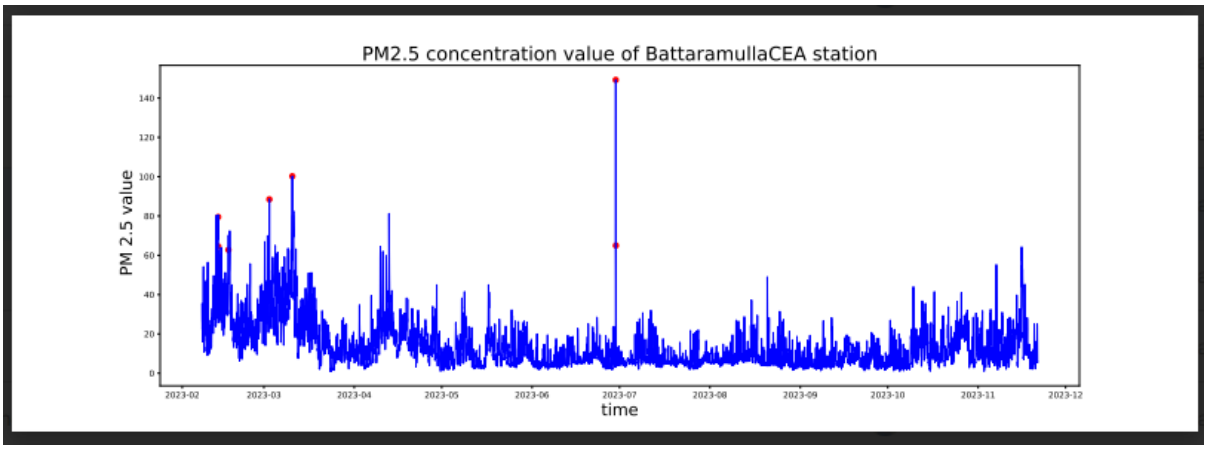


Figure 3.7: Detected outliers using the KNN method in Battaramulla

After identifying outliers next step is to choose a method to impute those outliers. For that several methods have been used,

1. Filling the outliers using the mean of the PM2.5
2. KNN imputation
3. Linear interpolation
4. Polynomial interpolation
5. Moving average method

To assess the efficacy of these methods, a randomly selected dataset is chosen, and outliers are imputed using each of the mentioned techniques. Subsequently, the Mean Square Error (MSE) is calculated for each method as a metric for evaluating imputation accuracy. The method yielding the lowest MSE is identified as the optimal approach for outlier imputation in our dataset.

	mean	KNN	linear	polynomial	moving avg	moving avg(3)
Ampara	41.75	0.12	10.35	6.11	16.89	11.10
Anuradhapura	45.91	0.21	8.00	12.84	7.62	8.00
BattaramullaCEA	39.71	0.35	4.95	4.23	10.44	4.95
Batticaloa	30.00	0.48	29.20	54.60	19.58	28.25
Fort	65.94	0.20	3.92	4.34	6.56	3.92
Galle	26.95	0.16	11.17	10.30	14.44	11.17
Hambanthota	6.79	0.04	0.30	0.10	0.66	0.30
Jaffna	10.79	0.21	0.48	0.87	1.43	0.48
Kandy	109.29	0.41	15.75	15.94	26.87	15.75
Kanthale	131.32	0.38	13.83	27.57	11.23	13.83
Katubedda	113.66	0.47	49.37	50.19	51.50	49.37
Kilinochchi	26.66	0.30	1.43	2.86	0.61	1.43
Mannar	14.73	0.20	4.54	3.90	5.79	4.60
Matara	39.73	0.13	7.80	13.02	8.95	7.80
Monaragala	14.05	0.35	0.99	1.22	1.08	0.99
Mullativu	33.75	0.51	9.32	9.81	9.63	9.32
PointPedro	8.99	0.31	3.92	8.42	1.92	3.92
Polonnaruwa	15.47	0.31	19.12	42.04	5.99	19.12

Figure 3.8: Data acquired using the KNN method in all stations

According to the figure above KNN imputation consistently shows superior results in terms of imputation accuracy. As a result, KNN imputation was used as the preferred approach for handling outliers in our PM2.5 dataset.

3.3.3 Preliminary analysis

3.3.3.1 Correlation between stations

Identifying correlations between stations allows us to understand the spatial relationships and dependencies among different monitoring locations. By quantifying the degree of correlation between PM2.5 concentrations measured at various stations, we can discern spatial patterns of pollution distribution. This knowledge is invaluable for clustering the stations.

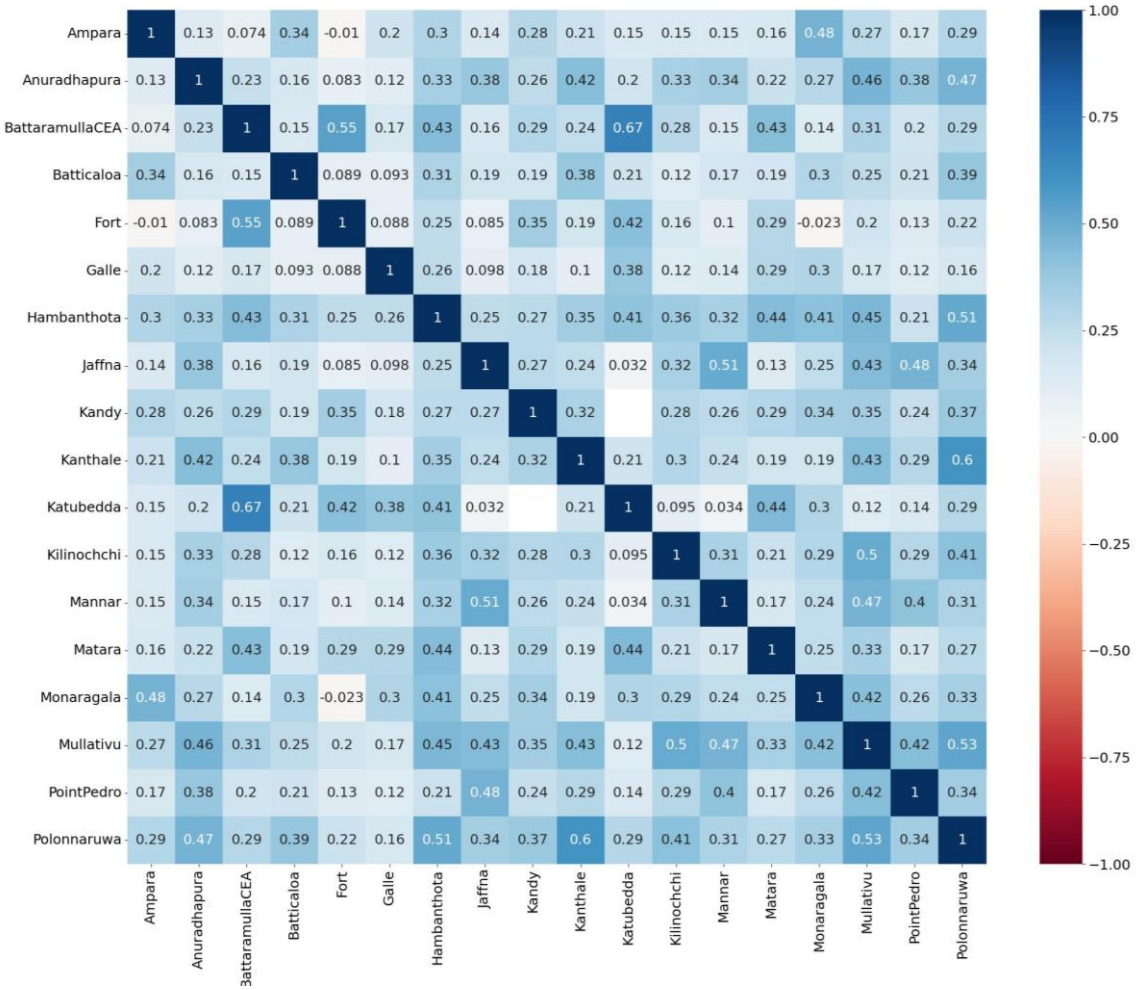


Figure 3.9: Correlation coefficients between the stations

The figure above shows the correlation coefficient values between different stations according to the above figure ‘Battaramulla’ and ‘Katubedda’ stations show the highest correlation coefficient value of 0.67. When analyzing the correlation coefficient values it can be observed that stations located closer to each other tend to have higher correlation coefficients. This can be due to the reason, stations that are close to each other often share similar environmental characteristics, such as local weather patterns, and topographical features.

3.3.3.2 Clustering the stations for modeling

Clustering the stations serves several important purposes in our research project on predicting PM2.5 concentrations. Since the lack of data in certain stations clustering the stations can used to increase the number of data points and it helps in mitigating overfitting, a common challenge in DL models.

By providing a larger and more diverse training dataset, clustering helps in regularizing the model and improving its ability to generalize to unseen data. This leads to DL models that are more robust and reliable in making predictions.

For clustering of the stations dendrogram chart was used.it is a hierarchical tree-like diagram commonly used in clustering analysis to visualize the arrangement of clusters and their relationships. In these algorithms correlation coefficient between stations is used for clustering the stations.

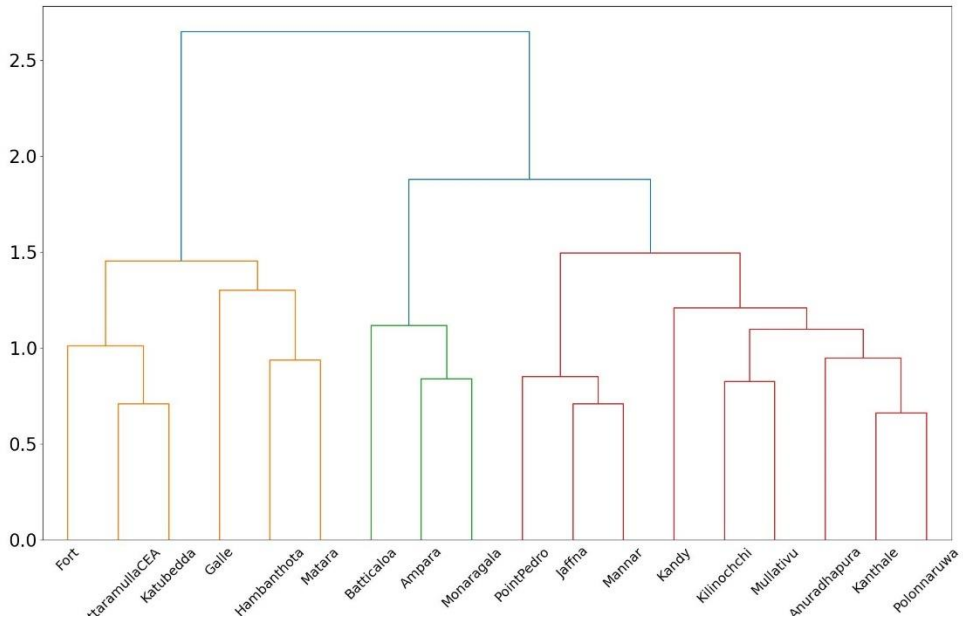


Figure 3.10: Dendrogram chart

In the above figure, each vertical line segment represents a cluster, and the height of the line indicates the distance at which clusters were merged. So, the clusters having low distance values have a high correlation. By defining the threshold values all stations can be divided into clusters. According to the above chart, all stations can be divided into 3 main clusters.

1. Cluster 1 - Ampara, Batticaloa, Ampara
2. Cluster 2 - Colombo Fort, Battaramulla, Katubedda, Galle, Hambanthota, Matara
3. Cluster 3 - Jaffna, Mannar, Kandy, Kilinochchi, Kanthale, Polonnaruwa, Anuradhapura, Mullaitivu, Point Pedro

3.4 Temporal Model development

We used a wide range of forecasting techniques during the model selection phase to determine the best method for predicting PM2.5 levels. In particular, we trained and implemented six different models: the Gated Recurrent Unit (GRU), the Long Short-Term Memory (LSTM), the Random Forest, and the Seasonal Autoregressive Integrated Moving Average (SARIMA), Exponential Smoothing (ETS), and the Autoregressive Integrated Moving Average (ARIMA). Because each of these models has distinct qualities and abilities, we are able to thoroughly assess how well they handle the intricate temporal dependencies present in PM2.5 data.

Exploring statistical and deep learning methods for PM2.5 prediction provides a more comprehensive approach. Statistical methods provide interpretability, robustness to noise, and computational efficiency, making them appropriate for understanding linear relationships in data and dealing with irregular real-world datasets. However, they may struggle with nonlinear dynamics, necessitating manual feature engineering. Deep learning methods excel at automatically capturing complex nonlinear relationships and long-term dependencies, but they are difficult to interpret, require significant computational resources, and may necessitate a large amount of training data. By combining the strengths of both approaches, we hope to improve the accuracy and robustness of PM2.5 prediction models while addressing their respective limitations.

Choosing the best model for PM2.5 prediction requires a thorough evaluation process. We start by defining relevant evaluation metrics and categorizing the dataset as training, validation, and test sets. Each candidate model is then trained and validated on the relevant datasets, with hyperparameters fine-tuned as necessary. The model with the best overall performance is chosen, taking into account predictive accuracy, interpretability, computational efficiency, and generalization to previously unseen data. Sensitivity analysis ensures that the chosen model is robust, while thorough documentation and reporting provide transparency and reproducibility. This systematic approach ensures that the selected model effectively addresses the challenges of PM2.5 prediction while also meeting the project's objectives.

3.4.1 Statistical Models

3.4.1.1 ARIMA Model

The Autoregressive Integrated Moving Average (ARIMA) model is a popular statistical method for forecasting time series data. It combines three main components to capture the temporal structure of the data.

- Autoregression (AR): This component represents the relationship between an observation and a series of lagged observations (its past values). The AR component of ARIMA represents the linear relationship between the current observation and its lag values. The 'p' parameter specifies the order of the autoregressive component, which determines how many lagged observations are included in the model.
- Integration (I): This component entails differencing the time series data to achieve stationarity. Stationarity indicates that the statistical properties of the series (such as mean and variance) remain constant over time. The 'd' parameter in ARIMA represents the amount of differencing needed to make the series stationary.
- Moving Average (MA): This component represents the relationship between an observation and a residual error obtained from a moving average model applied to lagged observations. The MA component of ARIMA captures short-term effects in the data, which helps to smooth out any random fluctuations. The 'q' parameter indicates the order of the moving average component, which determines how many lagged forecast errors are included in the model.

Finding the parameters (p, d, and q) for an ARIMA model usually starts with a visual inspection of the time series data to identify patterns and trends, followed by an examination of autocorrelation and partial autocorrelation plots to determine the values of p and q. The order of differencing (d) required to achieve stationarity is determined by examining the data's trend and seasonality.

While this process can be manual and iterative, built-in Python libraries like pmdarima provide automated methods for determining the optimal parameters. These libraries use advanced algorithms and statistical techniques to efficiently search through various parameter combinations and select the most appropriate.

3.4.1.2 SARIMA Model

The Seasonal Autoregressive Integrated Moving Average (SARIMA) model is an extension of the ARIMA model that incorporates seasonality into the forecasting process. In addition to the autoregressive (AR), differencing (I), and moving average (MA) components present in ARIMA, SARIMA also includes seasonal AR, seasonal differencing, and seasonal MA terms.

- Seasonal Autoregression (SAR): SAR terms capture the relationship between an observation and its lagged values at seasonal intervals. This accounts for the seasonal patterns and cyclicality present in the data.
- Seasonal Integration (SI): Similar to the non-seasonal differencing in ARIMA, seasonal differencing (SI) is applied to the time series data to remove any seasonal trends and achieve stationarity.
- Seasonal Moving Average (SMA): SMA terms model the relationship between an observation and the residual errors from the seasonal moving average model applied to lagged observations. This helps capture the short-term fluctuations and noise in the seasonal component of the data.

The seasonal components (P, D, Q) of SARIMA mirror the non-seasonal components (p, d, q) but apply to the seasonal variations in the data. Finding the optimal parameters for SARIMA involves a similar process of data analysis, visualization, and model evaluation as with ARIMA. However, SARIMA requires additional consideration of seasonal patterns and trends, making it particularly suitable for time series data with clear seasonal variations

3.4.1.3 ETS Model

Exponential Smoothing is a widely used method for time series forecasting that assigns exponentially decreasing weights to past observations. Unlike ARIMA or SARIMA models, Exponential Smoothing does not require the data to be stationary or possess explicit seasonality. The basic idea of Exponential Smoothing is to generate forecasts by combining the weighted sum of past observations with a smoothing parameter (alpha) that controls the rate at which the weights decrease exponentially. The most common form of Exponential Smoothing is Simple Exponential Smoothing, which is suitable for time series data without any trend or seasonality.

In Simple Exponential Smoothing, forecasts are generated by updating the exponentially the weighted average of past observations using the formula:

$$\hat{Y}_{t+1} = \alpha Y_t + (1-\alpha)\hat{Y}_t$$

$\hat{Y}_{t+1}$= Forecast for the next time period

Y_t = Actual observation at time period

$\hat{Y}_t$ = Forecast for time period

α is the smoothing parameter, also known as the smoothing factor or smoothing constant. It controls the rate at which the weights decrease, typically ranging from 0 to 1. One of the advantages of Exponential Smoothing is its simplicity and ease of implementation. It is computationally efficient and can handle time series data with irregular patterns or short term fluctuations. However, it may not perform well for data with complex trends or seasonality.

3.4.2 DEEP LEARNING MODELS

3.4.2.1 LSTM Model

Long Short-Term Memory (LSTM) is a recurrent neural network (RNN) architecture that can effectively model and predict sequential data, making it ideal for time series forecasting tasks. Unlike traditional feedforward neural networks, which process input data in a single pass, LSTM networks use feedback connections to retain and process information over time, allowing them to detect long-term dependencies in sequential data.

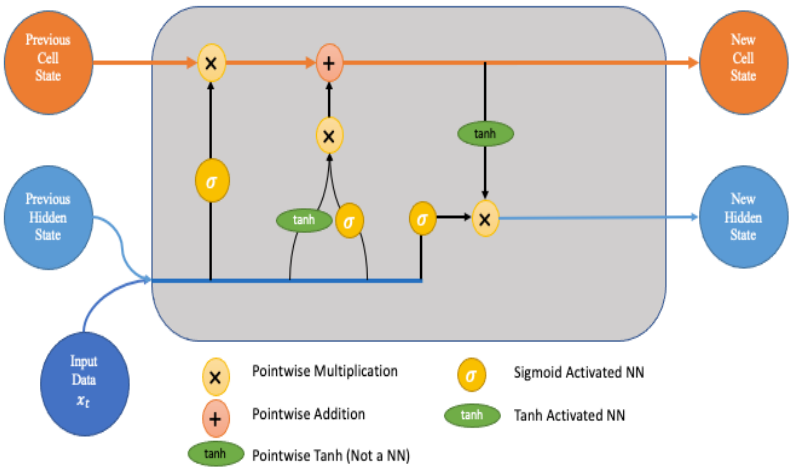


Figure 3.11: Long Short Term Memory Model Architecture

The key components of an LSTM network include:

- Memory Cells: LSTM networks contain memory cells that can store information over long periods of time. These memory cells are equipped with gating mechanisms that control the flow of information into and out of the cell, allowing the network to learn which information to remember or forget.
- Gates: LSTM networks have three types of gates: input gates, forget gates, and output gates.
  - Input gates regulate the flow of new information into the memory cell.
  - Forget gates control the retention or removal of information from the memory cell.
  - Output gates determine the output of the memory cell.
- Hidden States: LSTM networks maintain hidden states, which are updated based on the input data and the current state of the memory cells. These hidden states capture the network's learned representation of the input sequence and are used to make predictions.

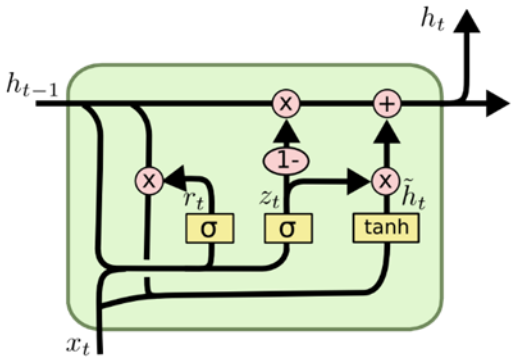
In the context of time series prediction, LSTM networks are trained using historical time series data to learn the underlying patterns and relationships. During training, the network adjusts its parameters to minimize the difference between the predicted values and the actual values in the training data. Once trained, the LSTM network can be used to generate forecasts by feeding it with historical data and predicting future values based on the learned patterns.

LSTM networks offer several advantages for time series prediction:

- Ability to Capture Long-Term Dependencies: LSTM networks can effectively capture complex temporal relationships and dependencies in the data, making them suitable for forecasting tasks with long-term patterns.
- Flexibility: LSTM networks can handle various types of time series data, including those with irregular patterns, trends, and seasonality.
- Automatic Feature Learning: LSTM networks can automatically learn relevant features from the input data, reducing the need for manual feature engineering.

3.4.2.2 Gated Recurrent Unit (GRU)

Gated Recurrent Unit (GRU) is a type of recurrent neural network (RNN) architecture, similar to LSTM, that is designed to effectively model sequential data, making it suitable for time series forecasting. GRU was proposed as a simpler alternative to LSTM, requiring fewer parameters and computational complexity while still capturing long-term dependencies in sequential data.



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Figure 3.12: GRU Model Architecture

- Update Gate: The update gate in GRU controls the flow of information from the previous hidden state to the current hidden state. It determines how much of the previous hidden state should be retained and how much of the new information should be incorporated.
- Reset Gate: The reset gate in GRU controls the extent to which the previous hidden state should be ignored when computing the new hidden state. It allows the network to selectively reset or update the information stored in the memory cells based on the current input.
- Hidden States: GRU networks maintain hidden states that represent the network's learned representation of the input sequence. These hidden states are updated based on the input data, the current hidden state, and the gates' activations.

In the context of time series prediction, GRU networks are trained using historical time series data to learn the underlying patterns and relationships. During training, the network adjusts its parameters to minimize the difference between the predicted values and the actual values in the training data. Once trained, the GRU network can be used to generate forecasts by feeding it with historical data and predicting future values based on the learned patterns.

GRU networks offer several advantages for time series prediction:

- Simplicity: GRU networks have a simpler architecture compared to LSTM, making them easier to train and interpret.
- Efficiency: GRU networks have fewer parameters and computational complexity, making them more computationally efficient, especially for smaller datasets.
- Flexibility: GRU networks can handle various types of time series data, including those with irregular patterns, trends, and seasonality.

3.4.2.3 RANDOM FOREST MODEL

Random Forest is a versatile and powerful machine learning algorithm that can be applied to various types of predictive tasks, including time series prediction. Here are some reasons why Random Forest is often used for time series prediction. Random Forest is an ensemble learning technique that generates predictions by combining several decision trees. Random Forest produces predictions that are more reliable and accurate by combining the predictions of several trees, which helps to lessen the variance and overfitting that are frequently associated with individual decision trees.

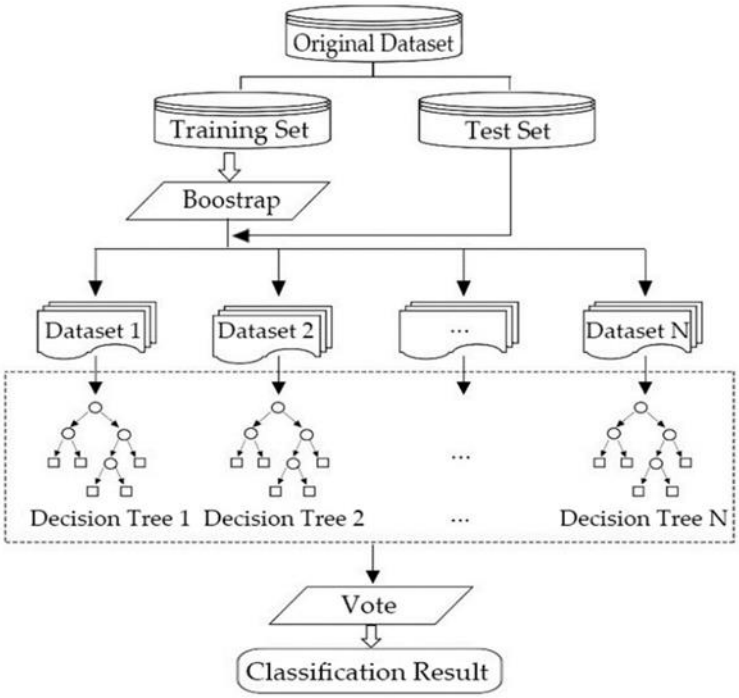


Figure 3.13: Random Forest Architecture

Feature Importance: Each input variable's contribution to the prediction is indicated by the feature importance metric that Random Forest offers. Time series prediction tasks can benefit from this as it helps determine which features are most important for predicting future values. Random Forest is inherently resistant to overfitting due to its ensemble approach and use of bootstrapping and feature bagging during training. This helps the model avoid learning noise or irrelevant patterns in the data, resulting in more generalizable predictions.

Handling Missing Values and Outliers: Random Forest can handle missing values and outliers in data without the need for imputation or preprocessing, making it ideal for real-world time series datasets with irregularities or measurement errors.

Scalability: Random Forest is relatively scalable and can handle large datasets effectively. It is easily parallelizable, resulting in faster training on multi-core processors or distributed computing environments.

3.5 CNN Model development

3.5.1 CNN (Convolutional Neural Network)

CNNs are a specific type of neural network architecture that is intended to process input in a topology resembling a grid. Because of this, they are especially well-suited to handle spatial data, such as pictures and videos, where there is a strong association between neighboring parts. So, because of that, we used CNN for spatial data. There are several key components of a CNN network.

- Convolutional layers - The fundamental units of CNNs are these filters (kernels), which are used to slide over the input data and apply convolution operations. Each filter is made to detect a particular pattern or feature, like corners, edges, or more intricate shapes in the case of deeper layers. As the filters move across the image, they produce a map that signifies the areas where those features were found.
- Pooling layers - Pooling reduces the dimensionality of feature maps, retaining the most essential information while reducing the computational load. This layer makes the model more robust to variations and distortions in the input. The pooling layer is used to reduce the number of parameters in the network and train the model in a fast manner resolving the overfitting problem of the model.

- Fully connected layer/Dense layer - Fully connected layers are one of the simplest types of layers in convolutional neural networks. Each and every neuron in a fully connected layer is fully connected to every other neuron in the previous layer. Fully connected layers are generally included in the last part of the CNN structure- when the objective is to leverage all the features that have been acquired by the convolutional and max pooling layers for making predictions.

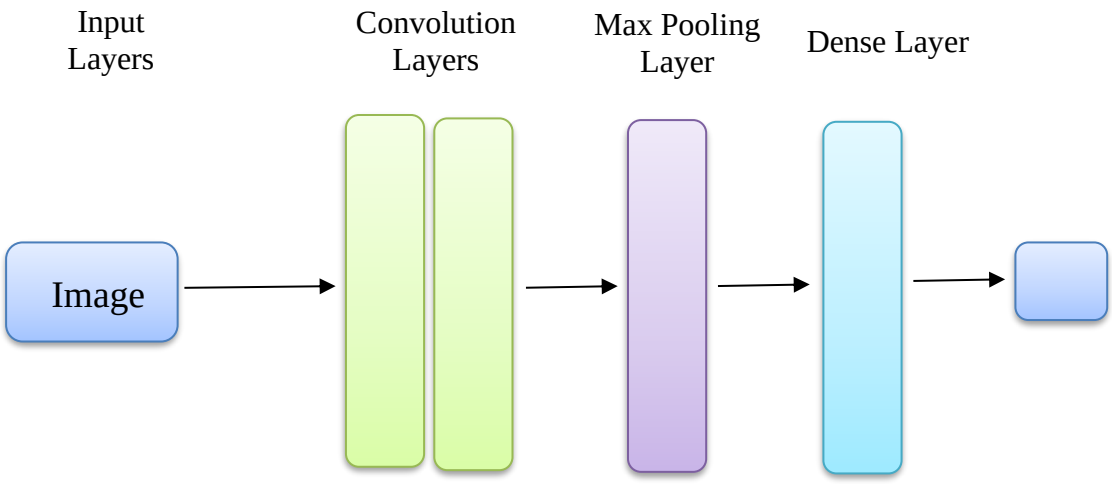


Figure 3.14: Basic components of a CNN Model

3.5.2 Transfer Learning

In machine learning and deep learning, transfer learning is the process of using a previously trained model as a jumping-off point for a new task that is linked to it. Through transfer learning, we may take advantage of the knowledge that the pre-trained model has already acquired rather than starting from scratch and training it on a large amount of data. A technique like this is useful when there is a limited number of data to train the model. For our task, we considered some of the most commonly used pre-trained deep learning models. There are VGG16, ResNet, and MobileNet.

3.5.2.1 VGG16

VGG16 is a type of convolutional neural network, and this is considered one of the best computer vision models to date. VGG-16 is renowned for its simplicity and effectiveness, as well as its ability to achieve strong performance on various computer vision tasks, including image classification and object recognition. This model is trained on the ImageNet data set which contains 14 million images belonging to 1000 classes. VGG16 consists of 16 layers, including 13 convolutional layers and 3 fully connected layers. Figure 3.15 shows the structure of the VGG16 model.

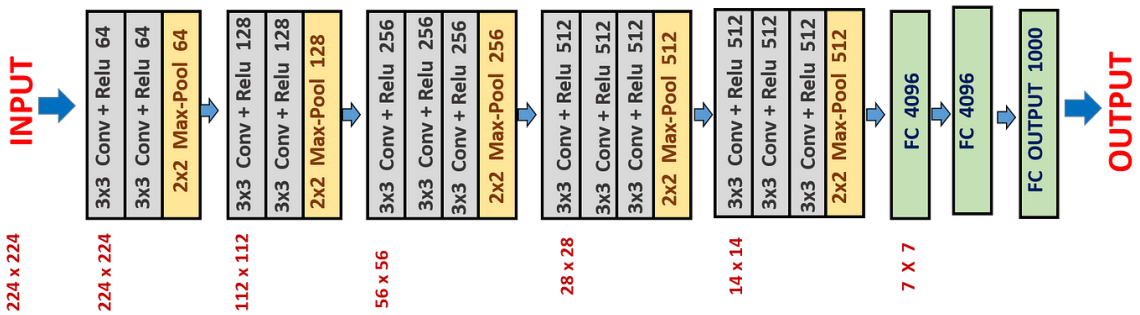


Figure 3.15: Structure of the VGG16 Architecture [10]

3.5.2.2 ResNet50

ResNet50 is another pre-train model used for image classification. There are 50 layers in ResNet50: 48 convolutional layers, 1 max-pooling layer, and 1 average pooling layer. Thanks to its residual connections, the design is deeper than previous networks like VGG16 without causing the problems that usually accompany greater depth. The vanishing gradient problem and other difficulties in training very deep networks were addressed by this architecture by utilizing "residual connections" or "skip connections," which are novel techniques, ResNet50 makes it possible for gradients to move across the network more readily during backpropagation.

The two main ResNet50 building blocks are the identification block and the convolutional block. The identity block is a basic block that adds the input back to the output after passing it through several convolutional layers. As a result, the network can learn residual functions, which convert input into desired output. The convolutional block is similar to the identity block but with the addition of a 1x1 convolutional layer that is used to reduce the number of filters before the 3x3 convolutional layer. Figure 3.16 shows the structure of the ResNet50 model.

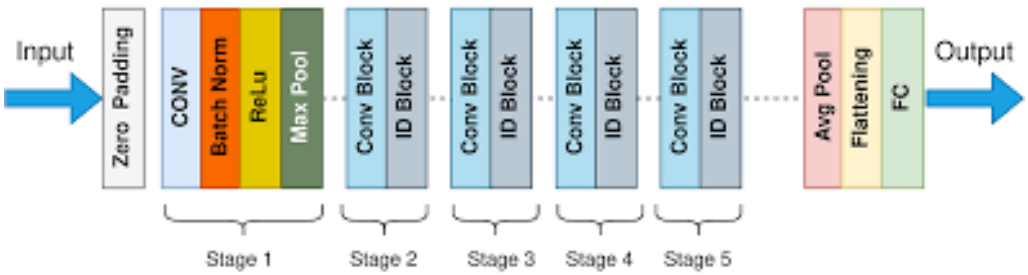


Figure 3.16: Structure of the ResNet50 Architecture [11]

3.5.2.3 MobileNet

MobileNet is an efficient CNN architecture that is used in real world applications. MobileNet mostly uses depth-wise separable convolutions rather than the typical convolutions used in previous architectures. Each depth-wise separable convolution layer consists of a depth-wise convolution and a point-wise convolution layer (Figure 04 shows the structure of a depth-wise separable convolution layer). Counting depth-wise and point-wise convolutions as separate layers, a MobileNet has 28 layers. In a depth-wise convolution, a single convolutional filter is applied to each input channel separately. The purpose of doing this is to reduce the computational cost by filtering each channel individually rather than combining them, as in a standard convolution. Following the depth-wise convolution, a point-wise convolution (1x1 convolution) is applied. This is used to mix the information from different channels of the depth-wise convolution layer.

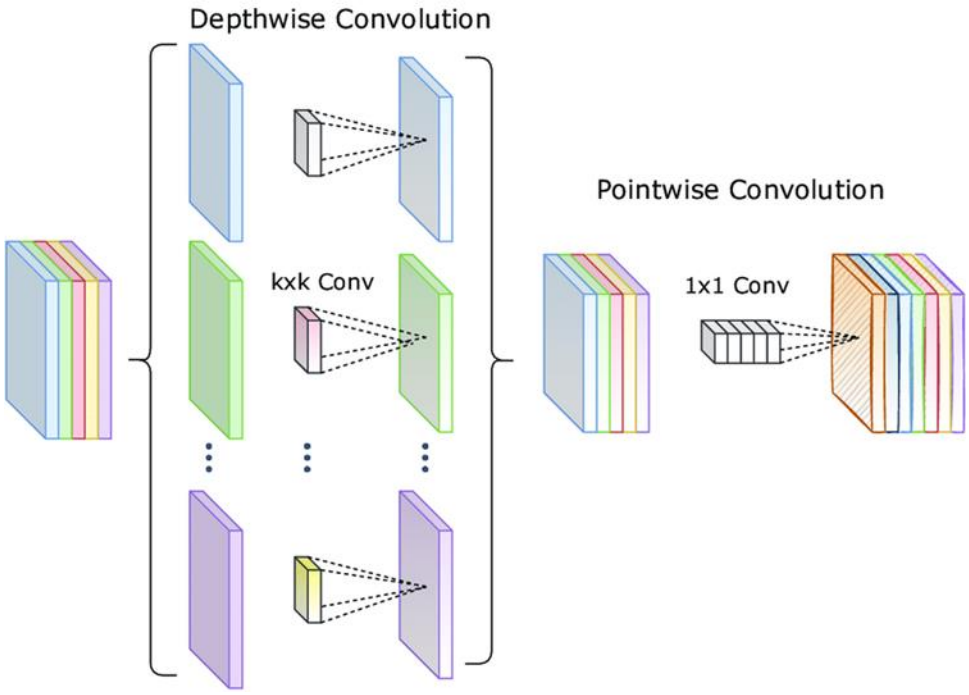


Figure 3.17: Depth-wise separable convolution layer [12]

A comparison of the results gained by these models can be seen in Chapter 4.5.3. The VGG16 model was the better among the models we used, and we decided to proceed with the VGG16 model as a pre-trained model.

3.5.3 Adding features

To improve the accuracy of our PM2.5 prediction model, we need to add more information to it than just the standard satellite picture inputs. Even while satellite photos offer useful spatial information, some environmental and contextual elements affecting air quality may be missed. We can more accurately depict the complex relationships between many variables, including greenery, human activity, and weather, and how these affect PM2.5 levels, by adding other features. By taking into account many aspects of the issue, these extra features enable the model to produce more accurate predictions, increasing its robustness and dependability in a variety of scenarios.

3.5.3.1 Green Index

The "Green Index" is one of the main features we are introducing. Satellite images with a high concentration of vegetation, especially trees, are associated with a higher density of green pixels, which is measured by the Green Index. Thanks to their natural ability to absorb and filter out airborne particles, trees significantly lower PM2.5 concentrations. We may consider how tree cover reduces air pollution by incorporating the Green Index into our model. Due to this feature, the model can distinguish between places with varied vegetation densities and modify its projections appropriately, producing a more accurate and nuanced picture of PM2.5 dispersion across various regions. These green regions are easily captured by defining an upper and lower bound of the green pixel range shown in Figure 3.18, the pixels which are falling to the in-between regions are considered green pixels then it is used to calculate the total pixel count.

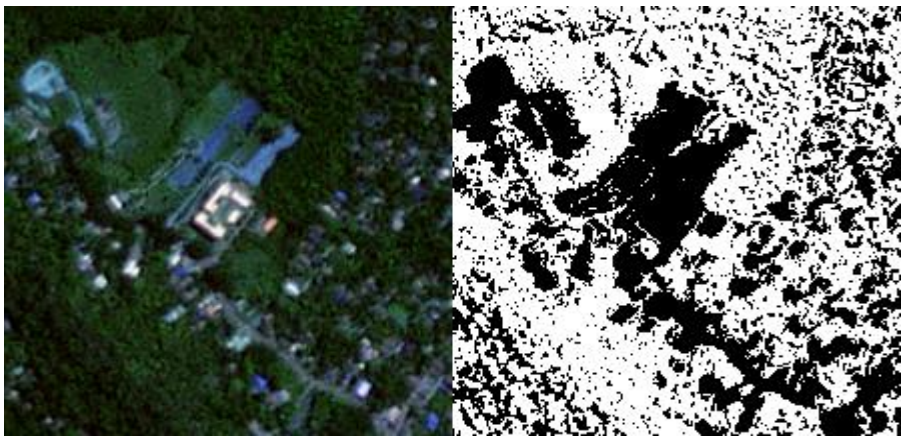


Figure 3.18: Results of the green index algorithm in Monaragala

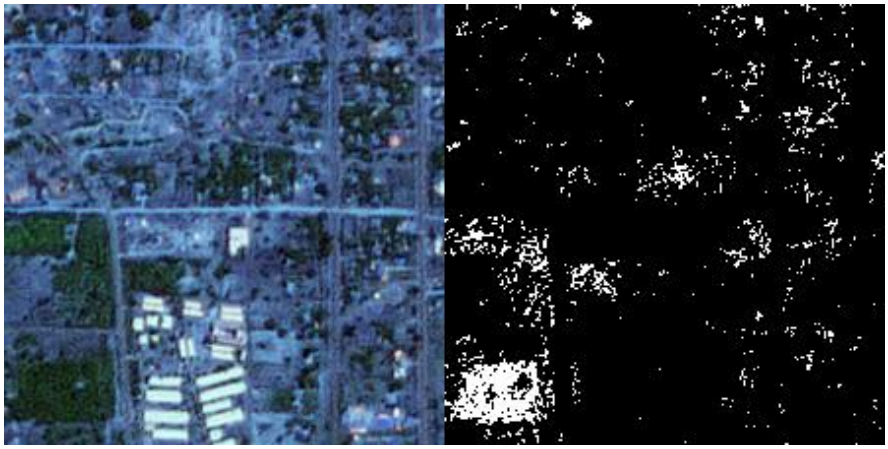


Figure 3.19: Results of the green index algorithm in Kilinochchi

3.5.3.2 Urban index

The second feature is the "Urban Index," indicating the measure of the intensity of brown and gray pixels appearing in satellite images as an approximation of how urbanized a particular area is. Urban areas are characterized by high levels of human activities, transportation systems, and industrial processes, which generally have higher levels of PM2.5 due to emissions from vehicles, factories, and other sources of pollution. The Urban Index quantifies the degree of urbanization in an area and can improve our forecasts regarding the levels of PM2.5. The computational algorithm for the derivation of the Urban Index is similar to that of the Green Index, in which distinct upper and lower thresholds are set for the brown and gray pixels. These are the characteristics typically associated with urban structures of buildings, roads, and other forms of infrastructure. This allows for more accurate delimitation of urban areas and assessments of their impact on air quality.

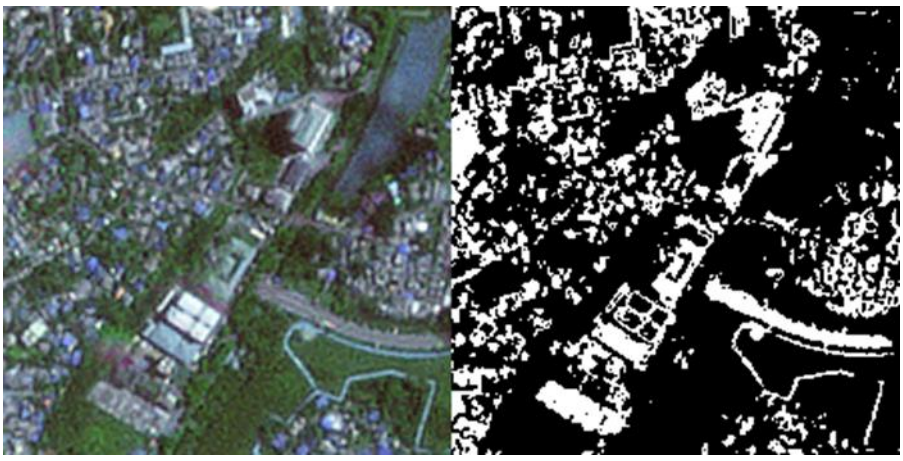


Figure 3.20: Results of the urban index algorithm in Battaramulla

3.5.3.3 Seasonal variations

The two most important monsoons in Sri Lanka are the Southwest Monsoon, generally from May to September, and the Northeast Monsoon, generally from December to February. Such marked seasonal variations influence the PM2.5 concentrations throughout the country accordingly.

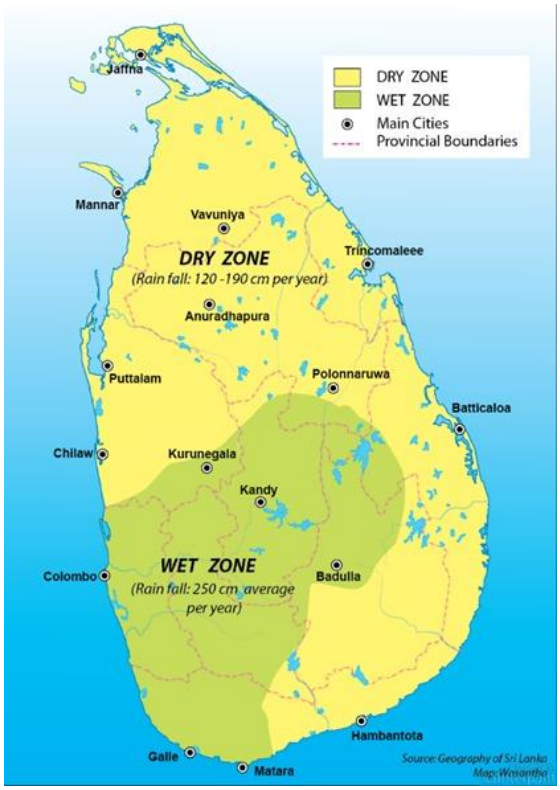


Figure 3.21: Dry zone and Wet zone of Sri Lanka

During the Southwest Monsoon, there is considerable precipitation over the west and southwest regions; in effect, this cleans the atmosphere, washing out airborne pollutants, and thereby PM2.5 concentrations are reduced. However, northern and eastern parts of the country have reduced rainfall in their desert areas, and the effect leads to high levels of PM2.5 since through wind action, dust, and other particulate matter are deposited. In the Northeast Monsoon, the northern and eastern states experience higher rainfall, thereby lowering PM2.5 concentration in those states, while the southern and western states will witness relatively drier weather. These seasonal trends create a difference in air quality between different regions; hence, seasonal variation is another variable for consideration in our model to predict the annual concentration of PM2.5.So there are 2 different regions can be identified; dry zone and wet zone, from the 18 monitoring stations, stations are located in the dry zone, and the wet zone is identified and input to the model as a feature.

3.5.3.4 Near Road

One of the main contributors to increasing PM2.5 concentrations is roads, especially in urban areas. Due to combustion that lets loose fine particulates into the atmosphere, vehicles on roads belong to the highest emitters of PM2.5. Besides, dust from the road, and wear of tires, and brakes are among the factors contributing to a high level of PM2.5 emissions. Areas close to highways show high values of these types of pollutant concentrations, which, in turn, impact the health conditions of the people residing nearby. The "Near Road" variable should be included in a model forecasting PM2.5, as it would allow the evaluation of the local impact of vehicular traffic on air quality. Determining the presence of a road in satellite imagery infuses the model with better capability to predict PM2.5 concentration in locations where traffic-related emissions are expected to be high. For the model, if a road is present in the satellite image True value is assigned to the Near Road feature for that monitoring station.

3.5.3.5 IsWeekday

This "IsWeekday" feature is supposed to capture the variation in PM2.5 concentrations that might change between weekdays and weekends. Normally, on weekdays, cities experience high human activities, which include increased traffic, industrial activities, and many other sources of pollution, thus leading to increased PM2.5. On the other hand, weekends normally experience reduced activities in commercial and industrial areas, which in turn leads to a reduction in PM2.5. This feature returns a binary value depending on whether the day the satellite image date falls is during the weekday.

Since there might be some seasonal pattern in the data where pollution cyclically follows, this division into two categories will help the model take into account such patterns in the data. Such a function includes simple steps: only an algorithm to check the date associated with each satellite image if it falls into the weekday or weekend category, and further, assign the binary value to it to include the temporal pattern in the PM 2.5 prediction process.

3.5.3.6 PM2.5 history

This is a critical enrichment of our model, whereby the "PM2.5 Lag History" variable incorporated 18 time-lagged values from a historical recording of PM2.5 obtained from the 18 monitoring stations. Such a feature brings the temporal dimension into the model to evaluate the effect of past pollution levels on current and future running concentrations.

Having these lagged values enables the model to learn temporal trends and patterns, making it better able to capture the persistence or attenuation of PM2.5 concentration. This historical context is of notable importance as air pollution typically presents temporal dependence; for instance, pollution conditions today are often influenced by those of the days before. Out of all the features implemented, PM2.5 Lag History appears to be the most efficient one since, through direct inclusion of empirical data into the predicting framework, it allows the model to produce much more accurate and reliable forecasts.

3.5.4 Final CNN model

When compared with other architectures such as ResNet50 and MobileNet, VGG16 was the most efficient base model, which is reflected from the evaluation done with different metrics: MAE, RMSE, R-squared, and MAPE (As shown in Table 4.2). The base model for constructing the final CNN model is taken as VGG16 architecture because, previously it has been shown to have the inherent capability to extract meaningful features from images. Additional features introduced are the Green Index, Urban Index, seasonal variations, Near Road, IsWeekday, and PM2.5 Lag History. All the features considered were targeted to include various environmental and temporal factors that have to do with air quality. The incorporation of these additional features with the robust VGG16 basic model achieved immense improvement in accuracy and reliability in the PM2.5 forecast, hence useful for air quality assessment. Figure 3.22 shows the final model for the spatial PM2.5 prediction model.

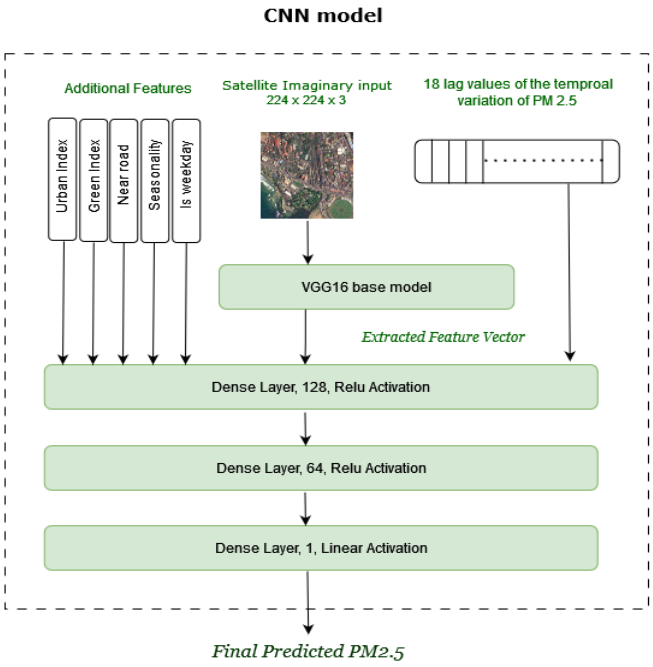


Figure 3.22: Final CNN model architecture

3.6 Hybrid Model development

A hybrid model that integrates both spatial and temporal data to improve the precision of PM2.5 level estimates. Our methodology blends CNN and LSTM network strengths to take use of various information sources. Spatial patterns and contextual information were captured by the CNN model using satellite pictures and manually derived environmental characteristics. Concurrently, to address the seasonal fluctuations and temporal dynamics affecting air quality, the LSTM model analyses historical PM2.5 data in conjunction with meteorological and temporal parameters. The hybrid approach increases PM2.5 forecast accuracy and expands its applicability to areas without a sensor network. Our methodology offers significant insights into air quality in situations when direct measurements are not possible by approximating PM2.5 levels in these neglected locations, hence facilitating more thorough environmental monitoring and decision-making.

To gather geographical patterns and contextual data on the air quality environment, the CNN component processes satellite images and manually extracts environmental variables like the urban index and green index. The comprehension of the spatial distribution and fluctuations in the vicinity of the sensor stations falls under the purview of this segment of the model.

On the other hand, the LSTM component is made to handle temporal data, such as temperature, relative humidity, historical PM2.5 values, day of the week, month, and hour of the day. It depicts the seasonal patterns and temporal dynamics affecting PM2.5 concentrations. In the combined architecture, the LSTM predictions are based on temporal patterns, while the CNN offers spatial estimations of PM2.5. LSTM model predicts PM2.5 levels for the current time point based on the temporal features. The CNN model provides PM2.5 estimates based on spatial features extracted from satellite images and manual features.

To combine these predictions effectively, we employ linear regression to determine the optimal weights for each model's output. This involves training the regression model on a validation dataset where both the CNN and LSTM predictions, along with actual PM2.5 values, are used. The regression model calculates the weights that minimize the prediction error, allowing us to balance the contributions of spatial and temporal insights. The final PM2.5 prediction is then derived by applying these weights to the CNN and LSTM outputs, ensuring that the integrated estimate leverages the strengths of both models for a more accurate and comprehensive air quality assessment. The total Hybrid model architecture is shown in Figure 3.23

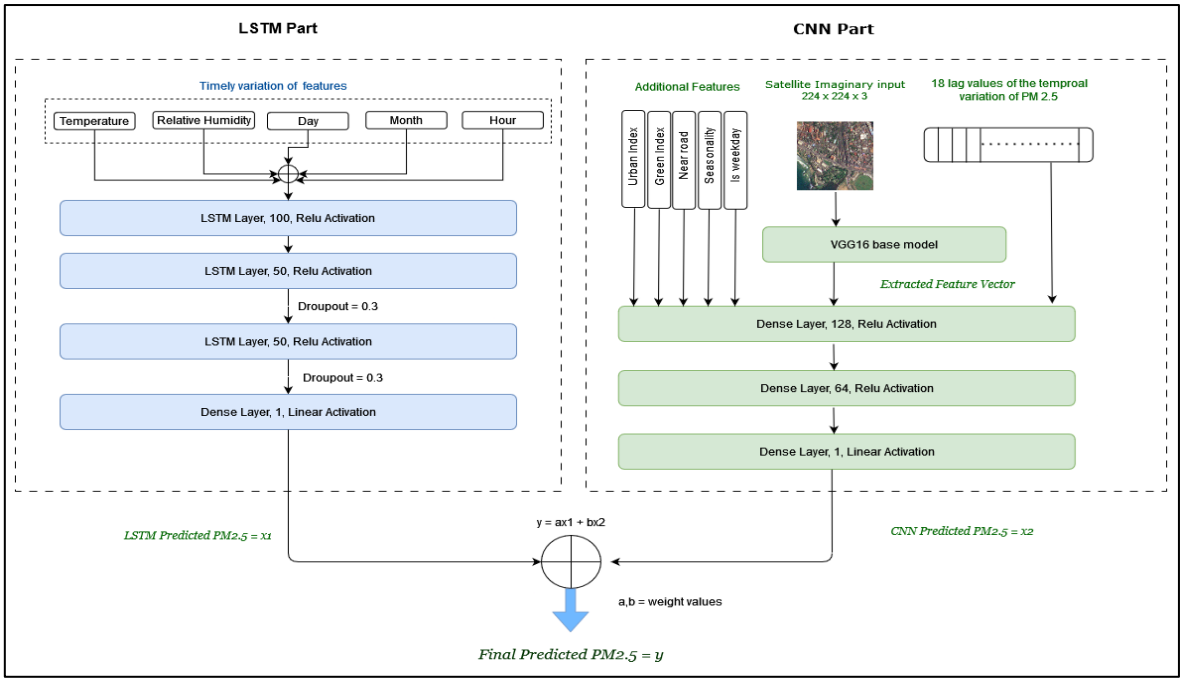


Figure 3.23: Hybrid Model Architecture

CHAPTER FOUR

RESULTS

4.1 Introduction

In this chapter, the results of the analysis, which include the performance evaluation of predictive models, and the accuracy of PM2.5 predictions, are included. As described in the previous chapter we used six predictive models which include three deep learning models and three statistical models. The results of each method were validated with the plotting graphs of test data and predicted values together in the same plot, and some validation methods like MAE, RMSE, and MSE have been used to validate the method. Through this brief analysis, we seek to understand air pollution dynamics in Sri Lankan urban cities.

4.2 Data Visualization

The data we used in this analysis contains an Hourly PM2.5 concentration of 18 stations which spans over nearly 7 months. Here we clustered these 18 stations into 3 clusters according to their geographic locations and correlation of their data. Following are the preprocessed data variations against the time of these 3 clusters. According to these three graphs, we can see that PM2.5 concentrations in cluster 1 vary between 3-25 ($\mu g/m^3$) in normal days. But some peak events have changed this concentration to around 30 ($\mu g/m^3$). According to the other 2 clusters, we can see that average values have been changed in some months. This may be monsoonal winds that occurred between December to May.

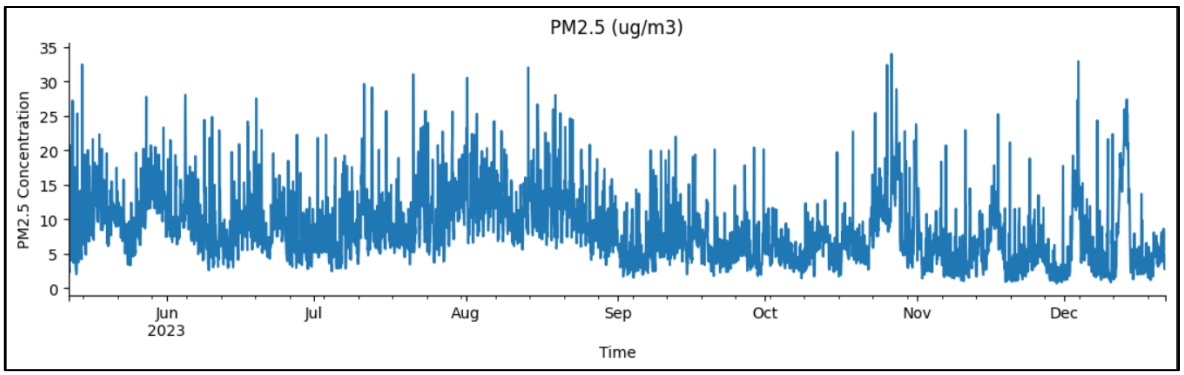


Figure 4.1: Hourly Concentration of PM2.5 ($\mu g/m^3$) in cluster 1.

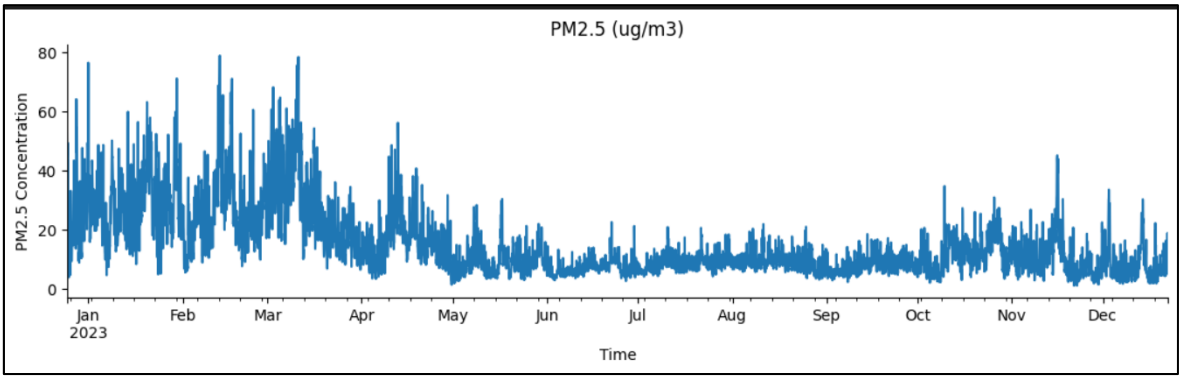


Figure 4.2: Hourly Concentration of PM2.5 ($\mu g/m^3$) in cluster 2

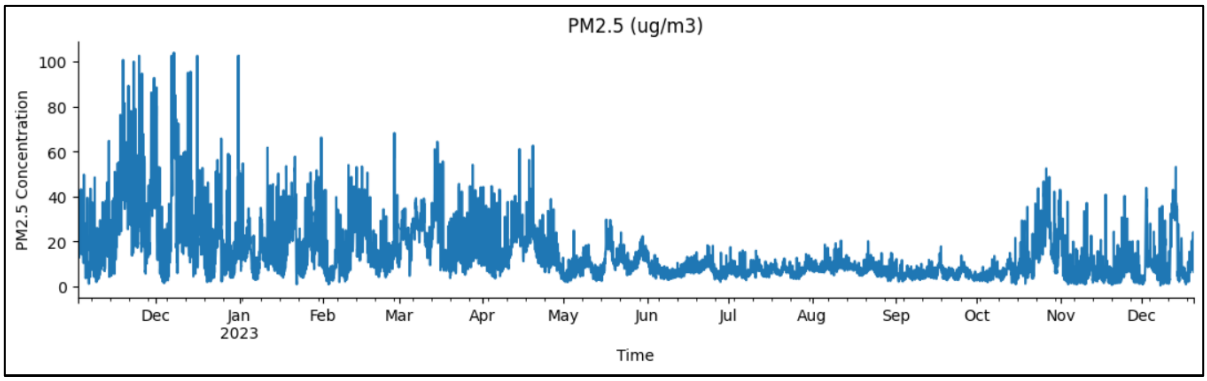


Figure 4.3: Hourly Concentration of PM2.5 ($\mu g/m^3$) in cluster 3.

4.3 Autocorrelation and Partial autocorrelation in the data

In this analysis, Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots are used to understand the autocorrelation structure of the data. The ACF plot shows the correlation between a series of data and its lagged values. Each point on the ACF plot represents the correlation between the series and its lagged values at different lags (time intervals). On the other hand, the PACF plot represents the correlation between the series and its lagged values after removing the linear dependence of the earlier lags. Similar to the ACF plot, each point on the PACF plot represents the correlation at different lags. These two plots help to identify the order of the ARIMA model by the p parameter can be identified using ACF plot while the q parameter can be identified using PACF plot. Following are the ACF and PACF plots of the 3 clusters.

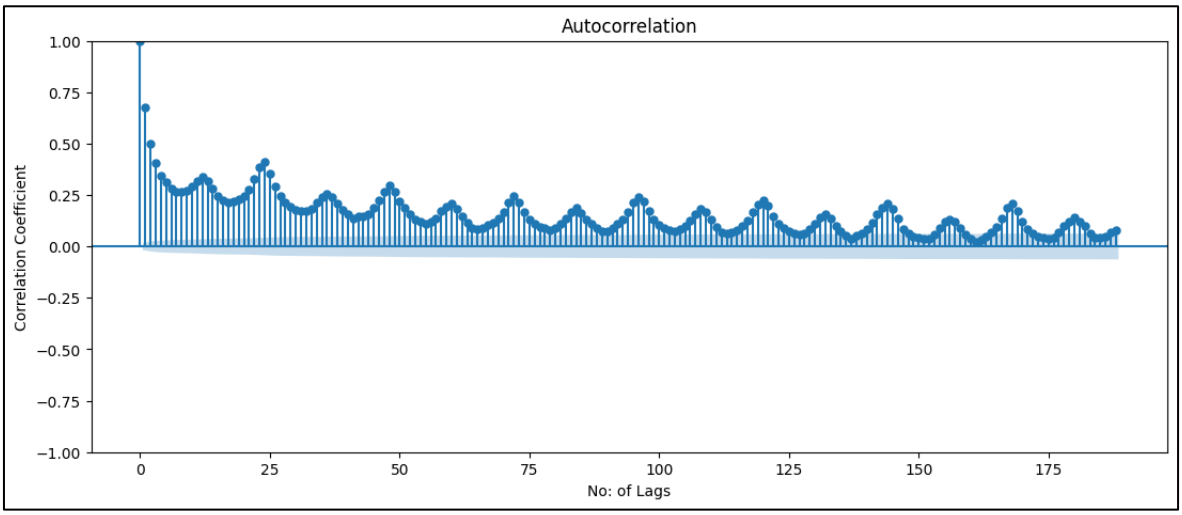


Figure 4.4: Autocorrelation of PM2.5 data in cluster 1.

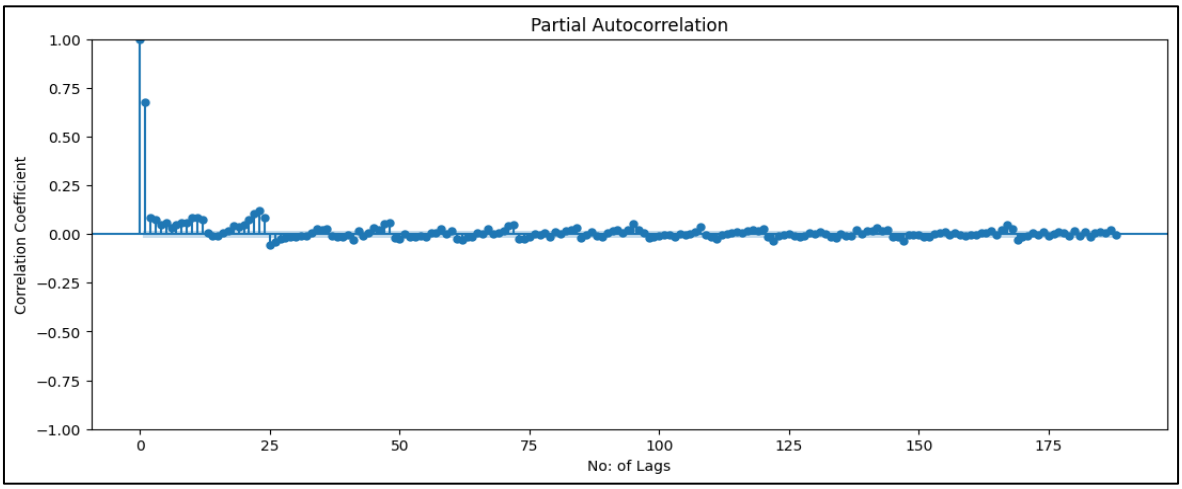


Figure 4.5: Partial autocorrelation of PM2.5 data in cluster 1

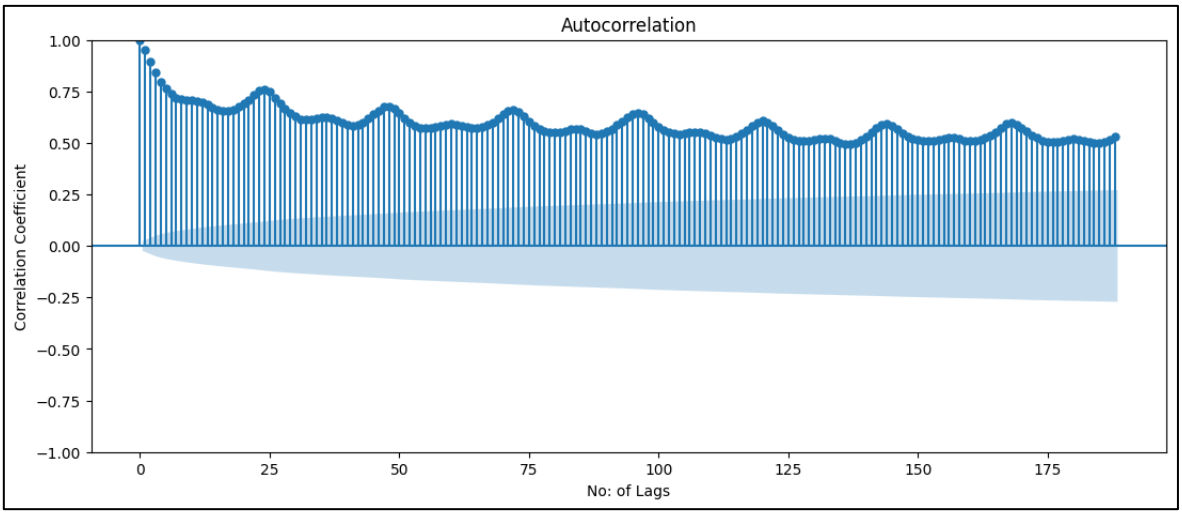


Figure 4.6: Autocorrelation of PM2.5 data in cluster 2

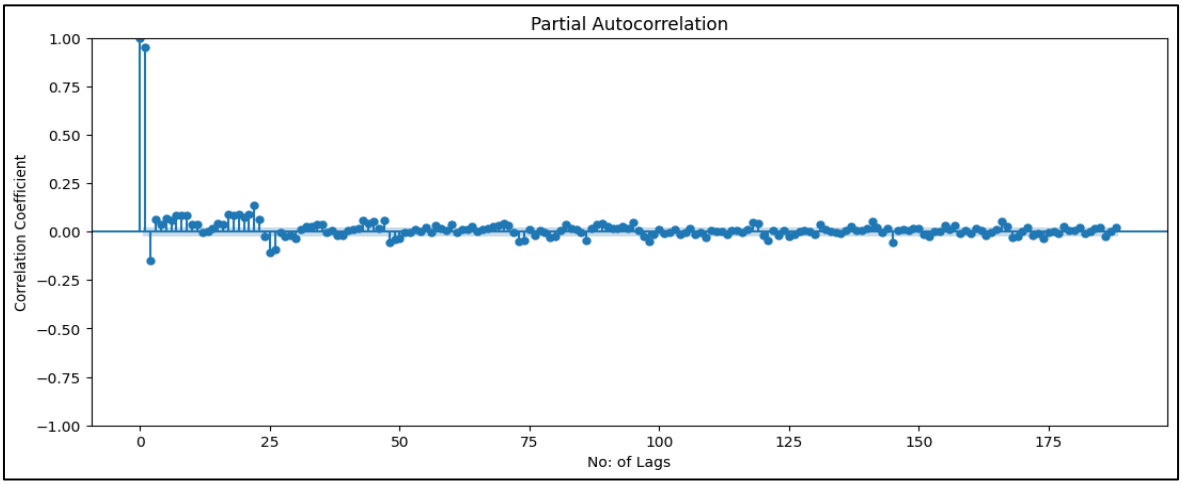


Figure 4.7: Partial autocorrelation of PM2.5 data in cluster 2

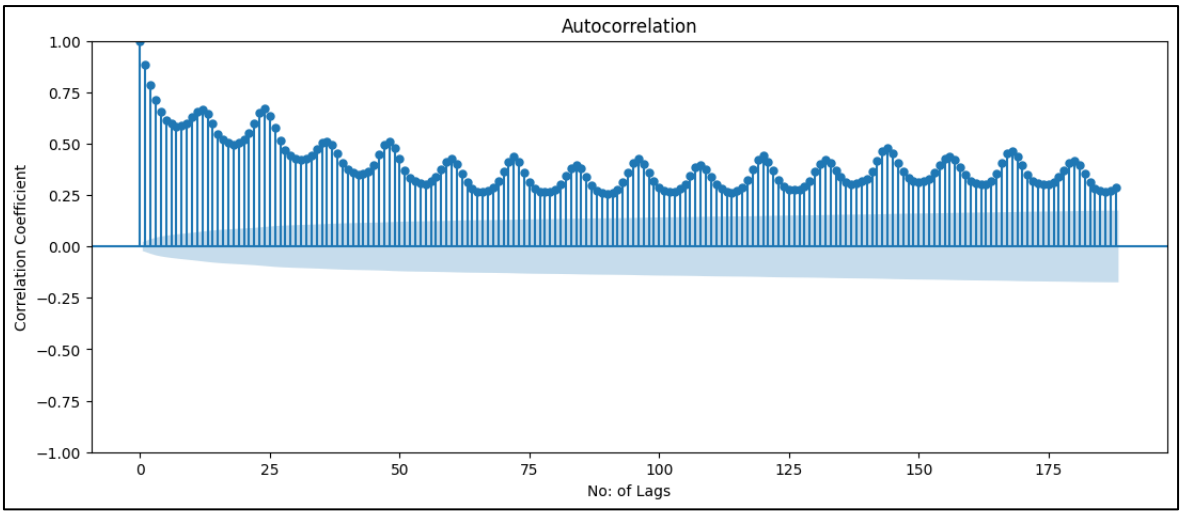


Figure 4.8: Autocorrelation of PM2.5 data in cluster 3

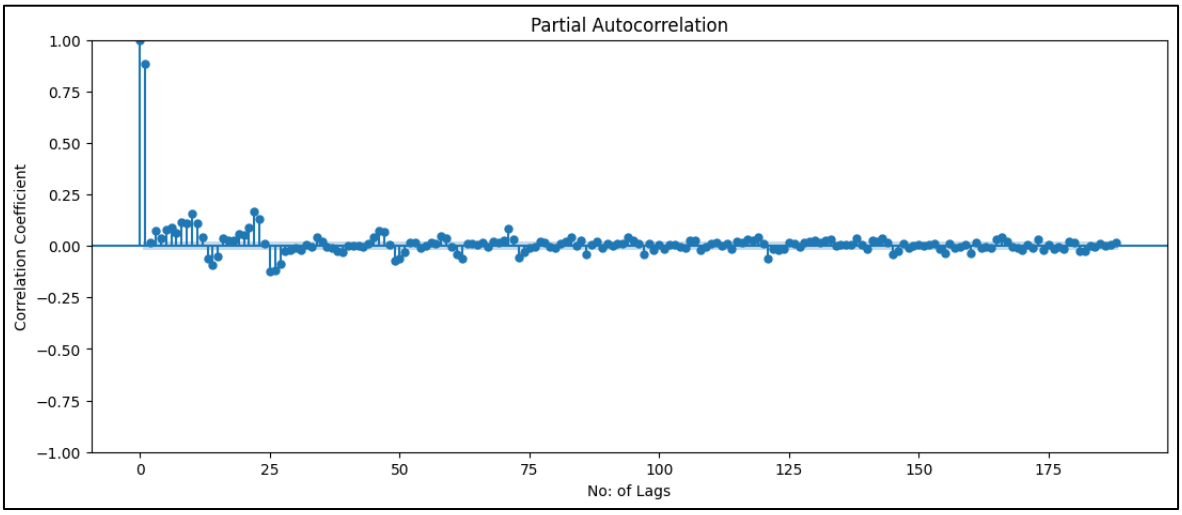


Figure 4.9: Partial autocorrelation of PM2.5 data in cluster 3

4.4 Temporal Model Performance Evaluation

In this section, we evaluate the performance of our temporal predictive models by comparing the predicted values generated by the models with the actual test values obtained from the dataset. The accuracy of our models is compared through visual representations to check how well our models capture the temporal trends and variations in PM2.5 levels.

4.4.1 ARIMA Model

ARIMA Model is a simple linear model used in time series predictions. Following are the results obtained through that model.

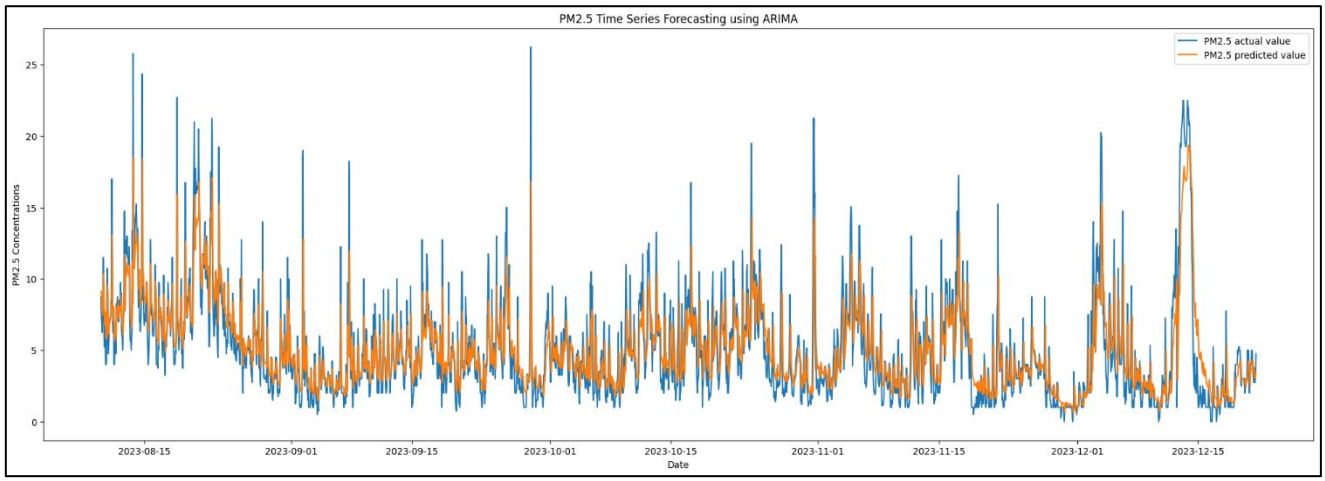


Figure 4.10: Predicted values of ARIMA Model vs Test data values of cluster 1

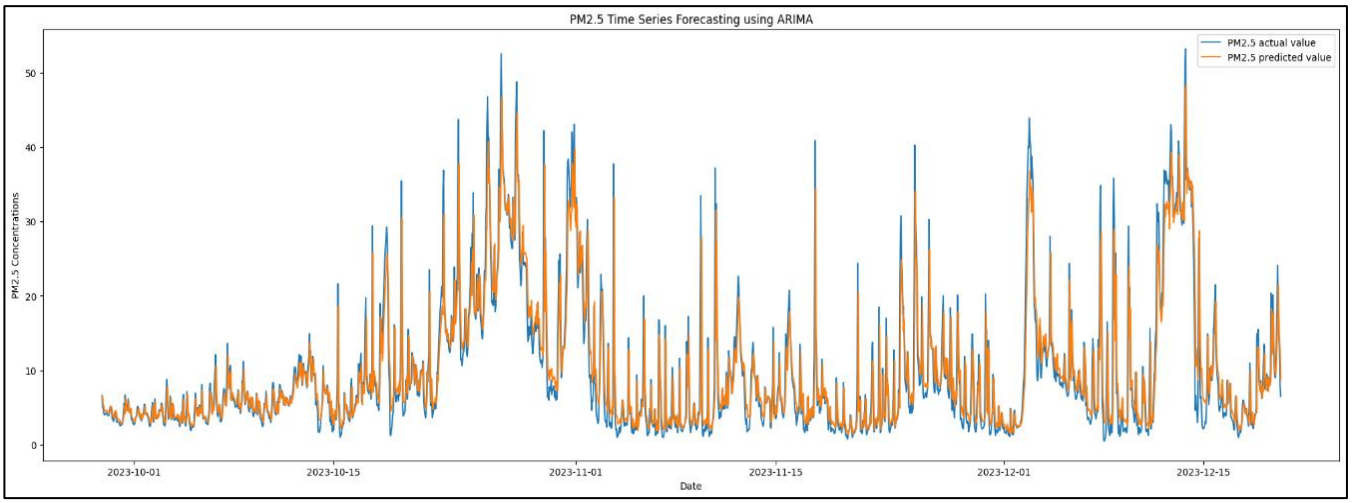


Figure 4.11: Predicted values of ARIMA Model vs Test data values of cluster 2

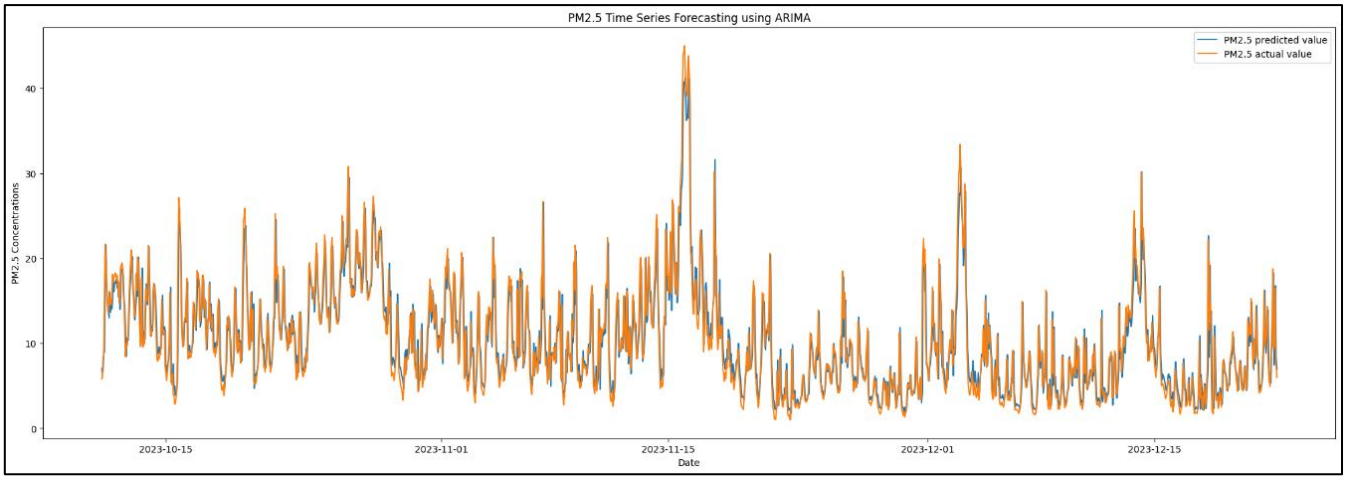


Figure 4.12: Predicted values of ARIMA Model vs Test data values of cluster 3

The graphs illustrate the comparison between the predicted PM2.5 values generated by the ARIMA model and the actual observed values from the test dataset. Overall, the ARIMA model demonstrates a close alignment between the predicted and actual PM2.5 concentrations, indicating its ability to capture the underlying patterns and fluctuations in the data. However, there are instances such as peak events where the predicted values deviate slightly from the observed values. We should try deep learning methods to get rid of those deviations.

4.4.2 SARIMA Model

The SARIMA Model is an extension of the ARIMA model while this model has an additional component related to the seasonality of the data.

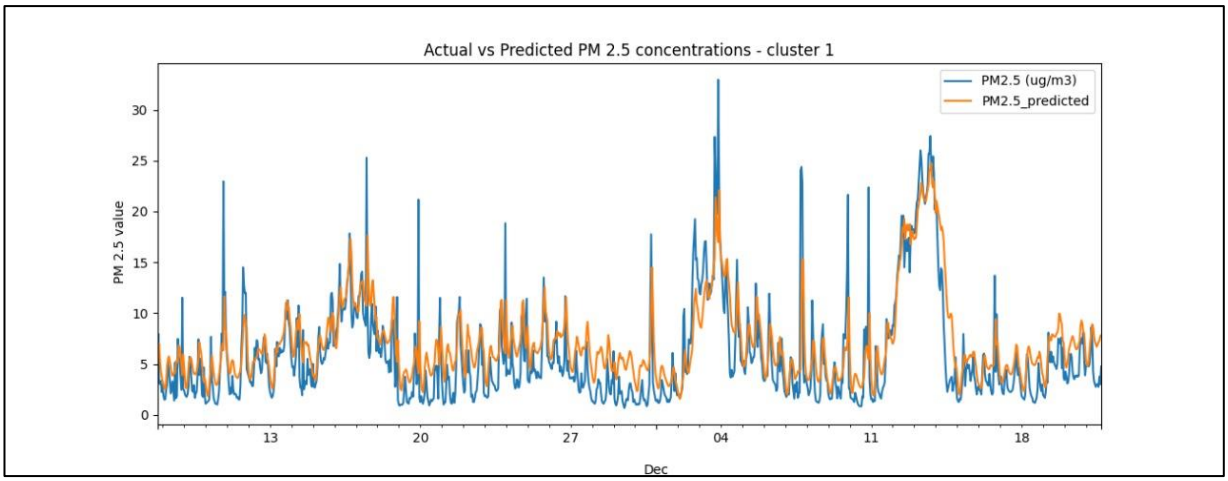


Figure 4.13: Predicted values of SARIMA Model vs Test data values of cluster 1

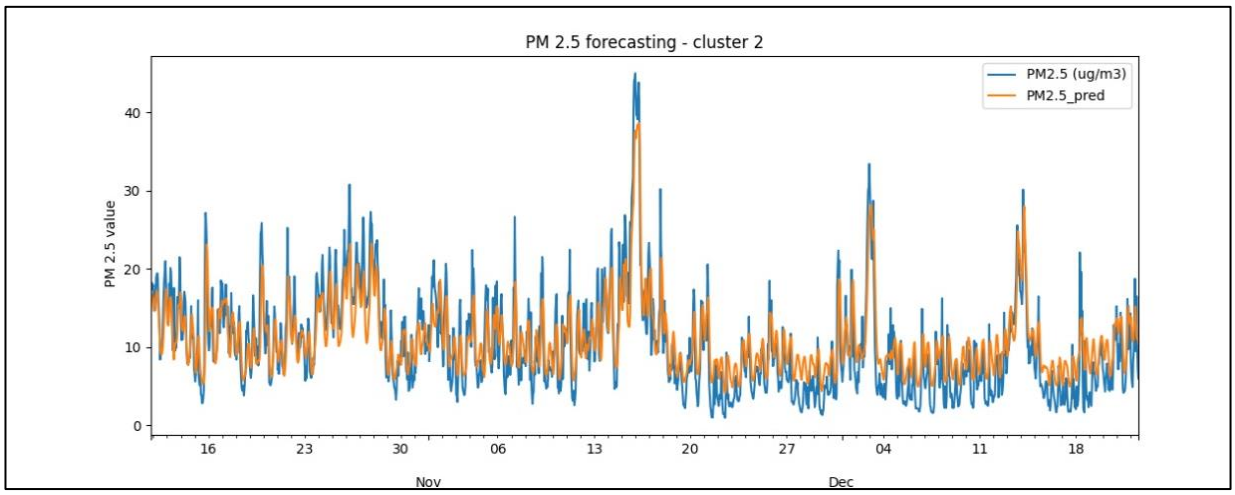


Figure 4.14: Predicted values of SARIMA Model vs Test data values of cluster 2

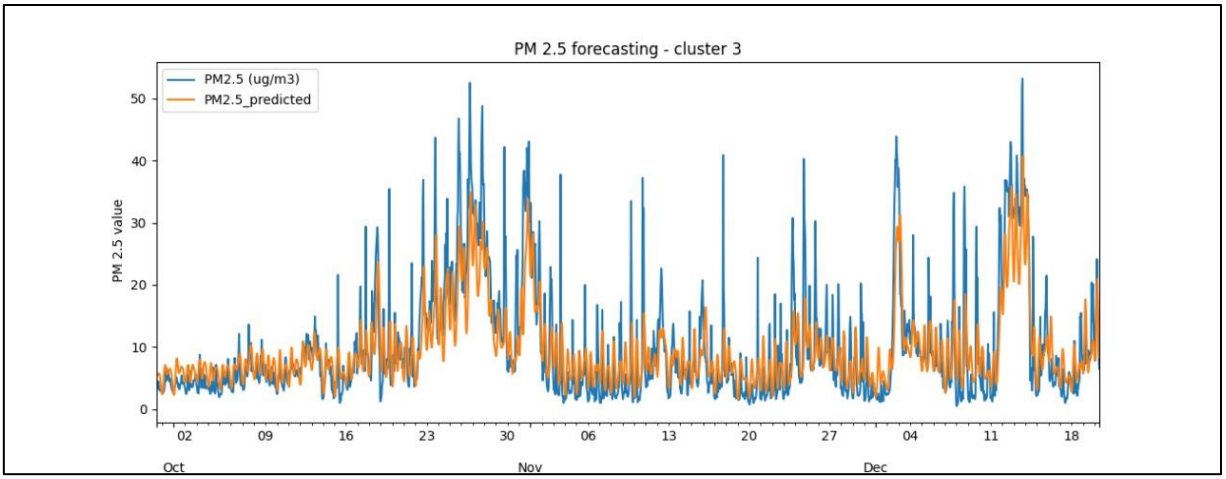


Figure 4.15: Predicted values of SARIMA Model vs Test data values of cluster 3

The graphs illustrate the comparison between the predicted PM2.5 values generated by the SARIMA model and the actual observed values from the test dataset. Here in this data, we could not find the seasonality in our data set. Because we haven’t enough data to calculate the yearly seasonality of the dataset. So this Model doesn't give much accuracy than the ARIMA model as we expected. So this model doesn’t fit our data well.

4.4.3 ETS Model

The Exponential Smoothing State-Space (ETS) model, a time series forecasting technique, uses exponential smoothing to capture underlying patterns and trends in sequential data, this helps to identify seasonality variations in time series data. However, we found that our data doesn't have any monthly seasonal variations. So this model also doesn't give a good prediction as we expected. As we can see in the graphs this model doesn't understand the data patterns in our data. So predicted values have big deviations from the test data values. So we would use a deep learning model instead of these models.

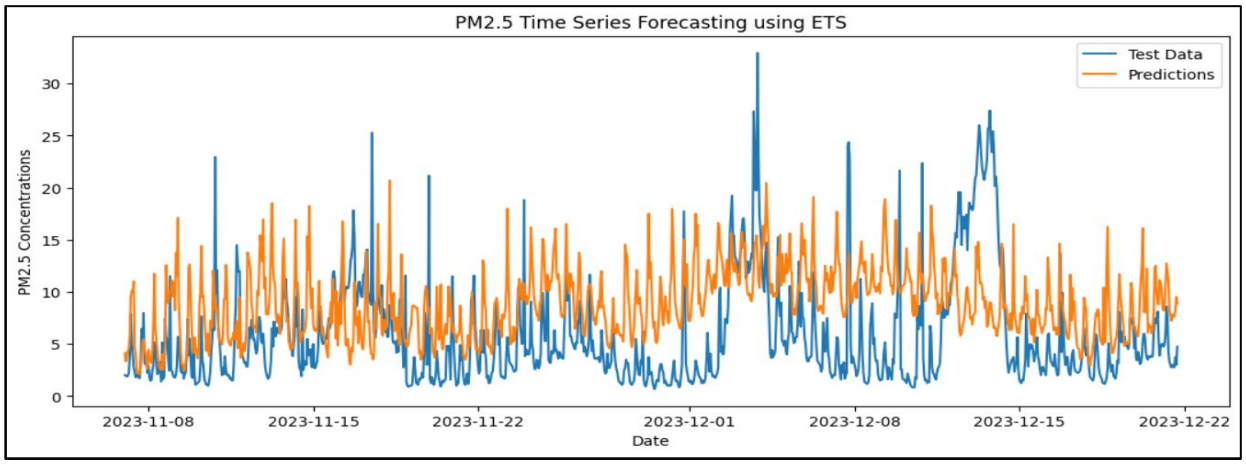


Figure 4.16: Predicted values of ETS Model vs Test data values of cluster 1

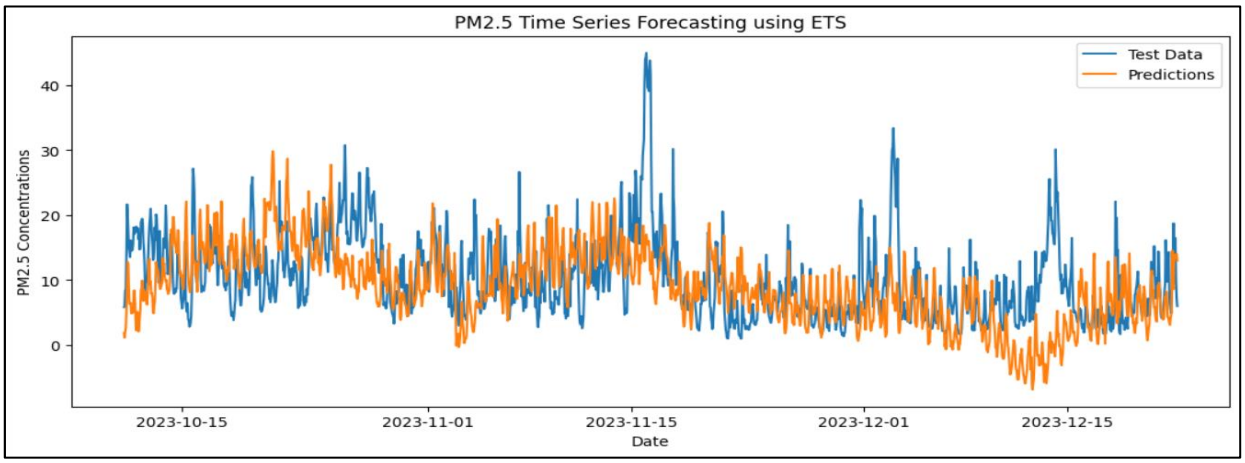


Figure 4.17: Predicted values of ETS Model vs Test data values of cluster 2

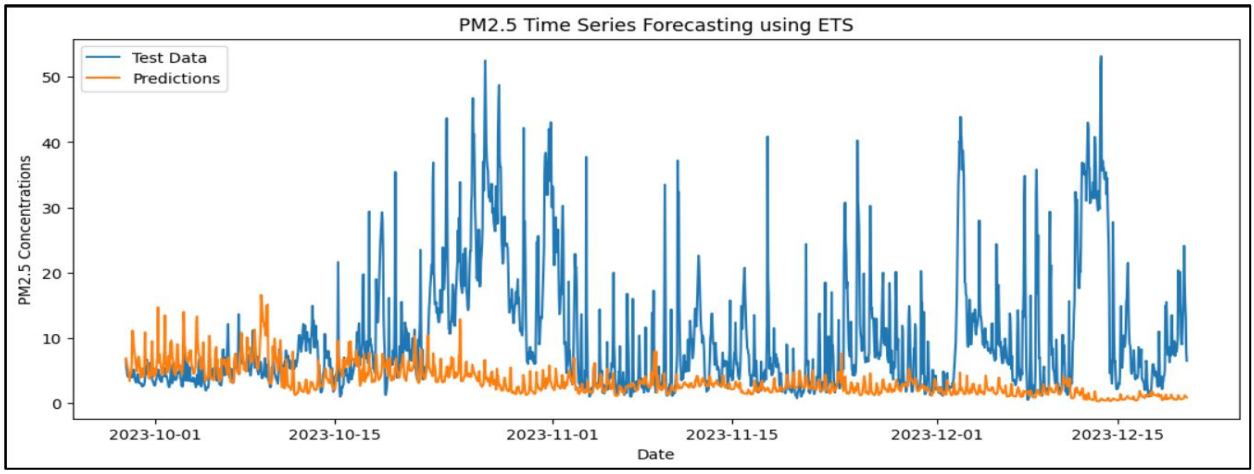


Figure 4.18: Predicted values of ETS Model vs Test data values of cluster 3

4.4.4 LSTM Model

The Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture. LSTMs are capable of learning and remembering information over long sequences, making them well-suited for time series forecasting. Looking at the graphs this method is very good at predicting future values. Both the test values and predicted values are the same in most of the cases.

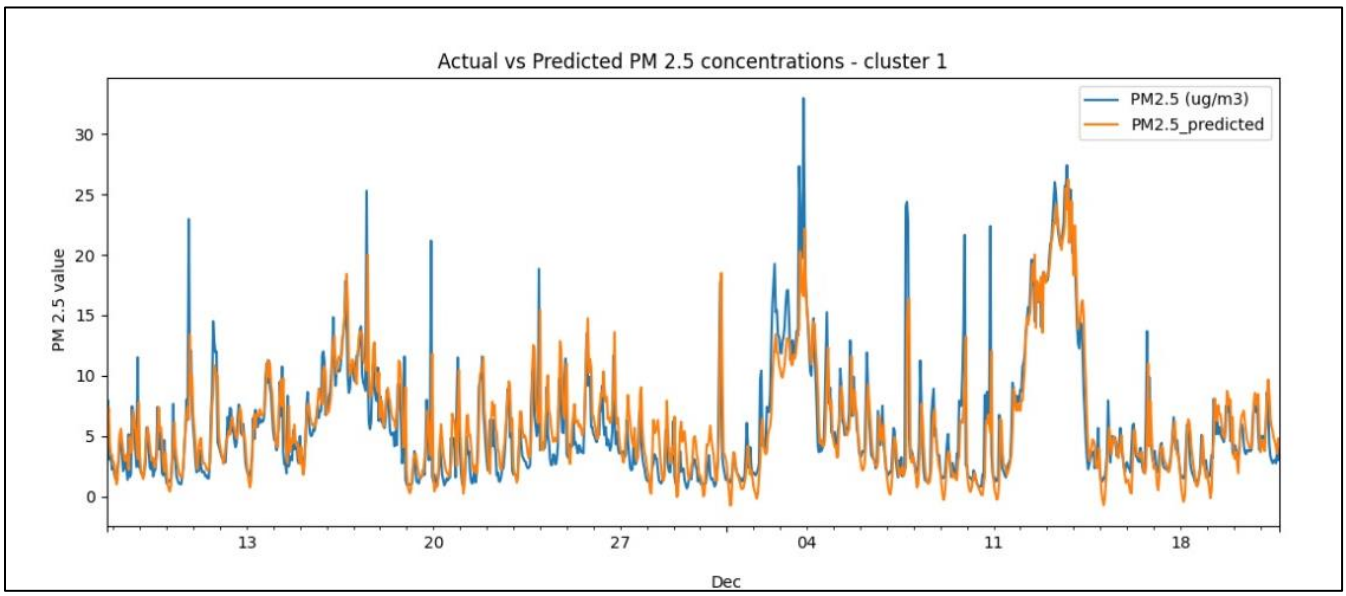


Figure 4.19: Predicted values of LSTM Model vs Test data values of cluster 1

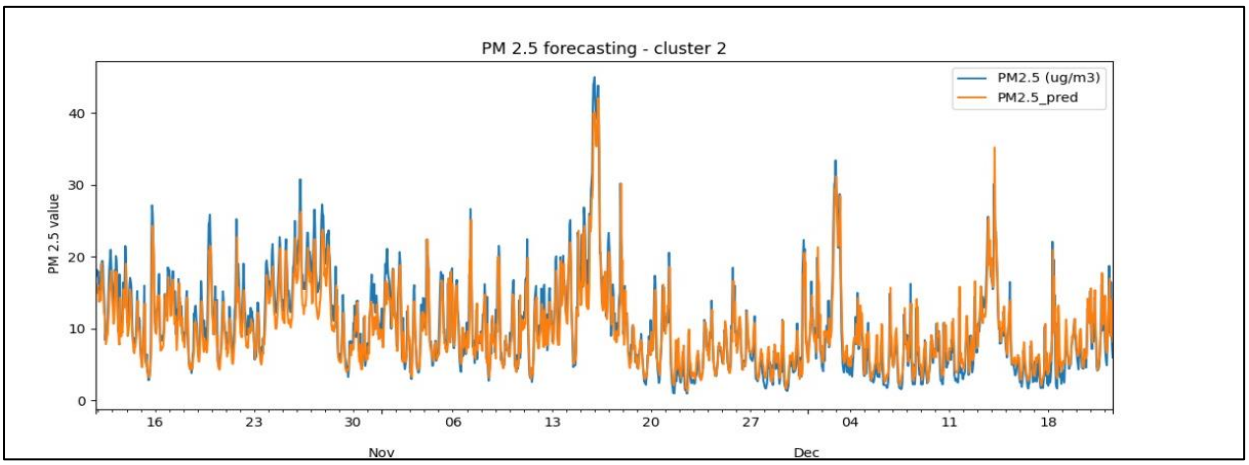


Figure 4.20: Predicted values of LSTM Model vs Test data values of cluster 2

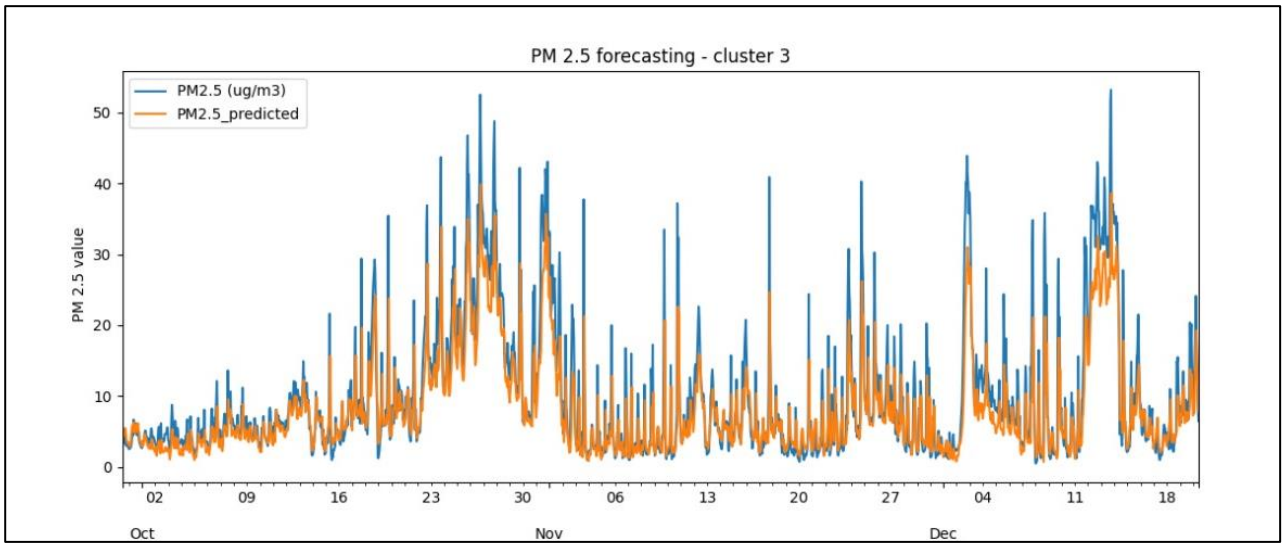


Figure 4.21: Predicted values of LSTM Model vs Test data values of cluster 3

4.4.5 GRU Model

TheGated Recurrent Units (GRUs) are a type of recurrent neural network (RNN) architecture similar to Long Short-Term Memory (LSTM) networks. They are designed to capture and model sequential data, such as time series, by efficiently learning long-term dependencies. However, GRUs have a simpler architecture compared to LSTMs, with fewer parameters and computational requirements. However, as our data set is small this has been more effective than the LSTM model. But looking at the graphs of LSTM and GRU we can see like same accuracy has been in both methods.

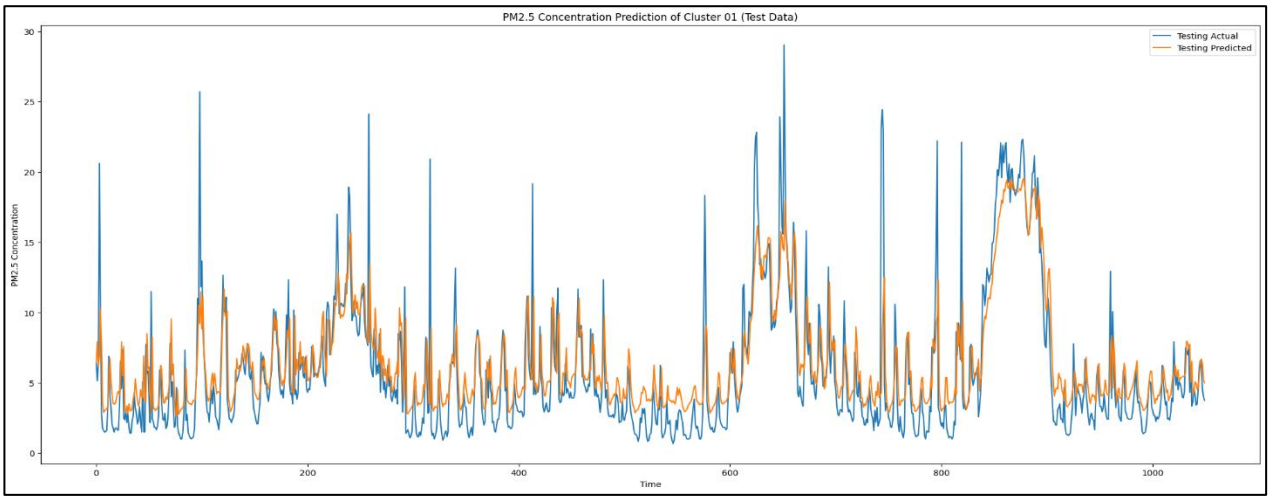


Figure 4.22: Predicted values of GRU Model vs Test data values of cluster 1

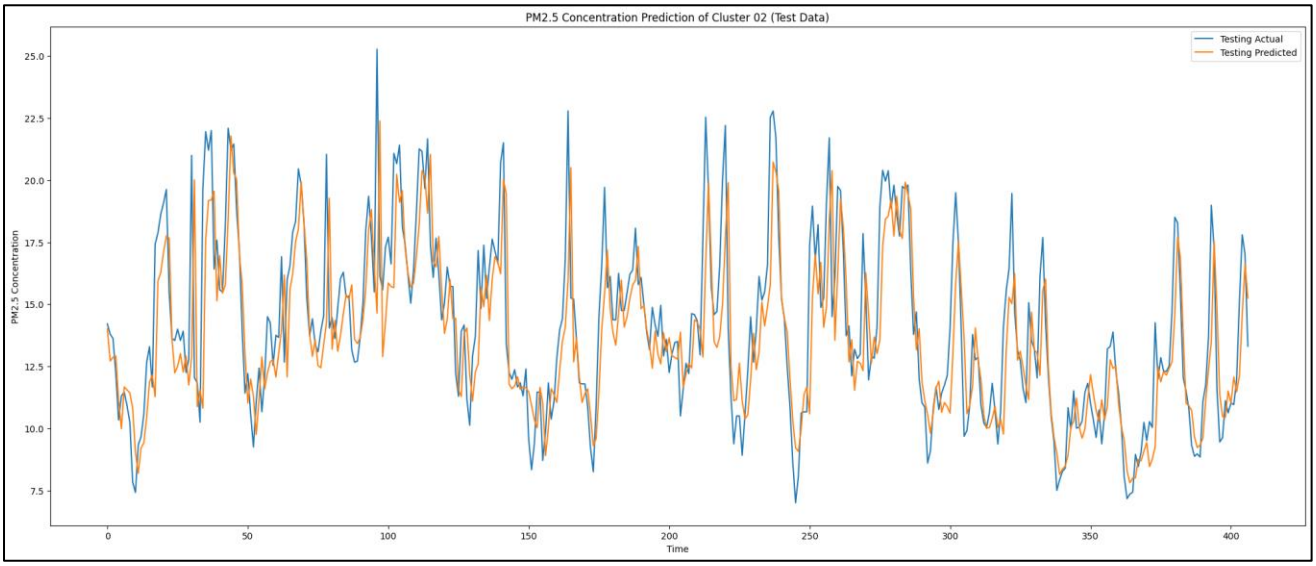


Figure 4.23: Predicted values of GRU Model vs Test data values of cluster 2

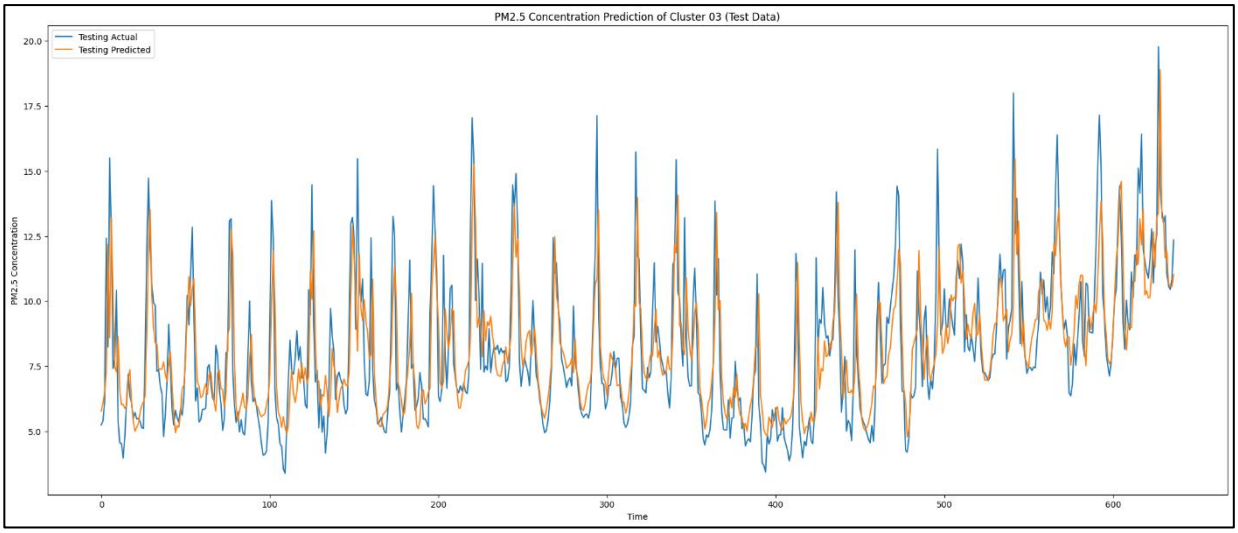


Figure 4.24: Predicted values of GRU Model vs Test data values of cluster 3

4.4.6 Random Forest Model

The Random Forest is a popular machine-learning algorithm used for both classification and regression tasks. It works like a forest with a lot of trees and branches. The branch with the higher probability is taken as the predicted value. But for our dataset, this was not much more accurate than the previous methods like LSTM and GRU. As seen in the graph this model couldn't fit our test data well. But it somehow nearly caught the pattern of our dataset.

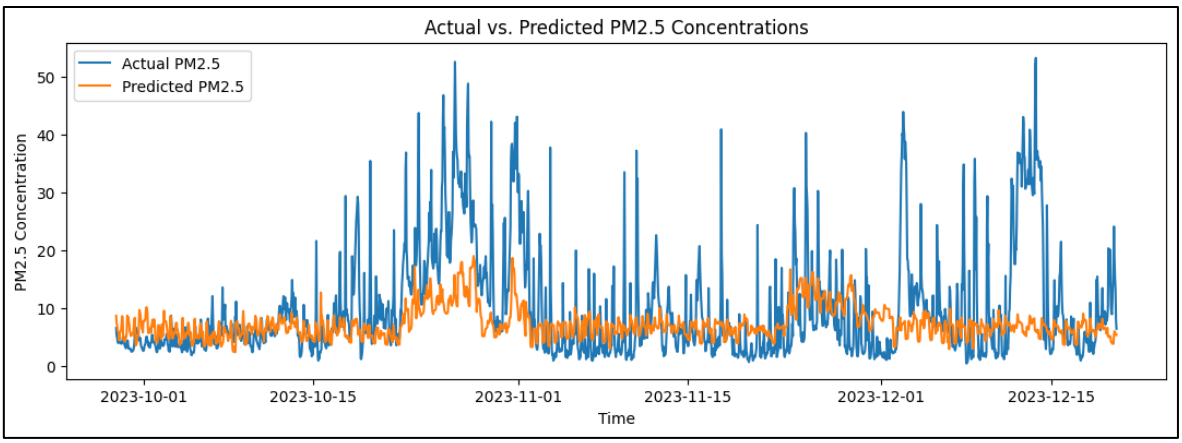


Figure 4.25: Predicted values of Random Forest Model vs Test data values of cluster 1

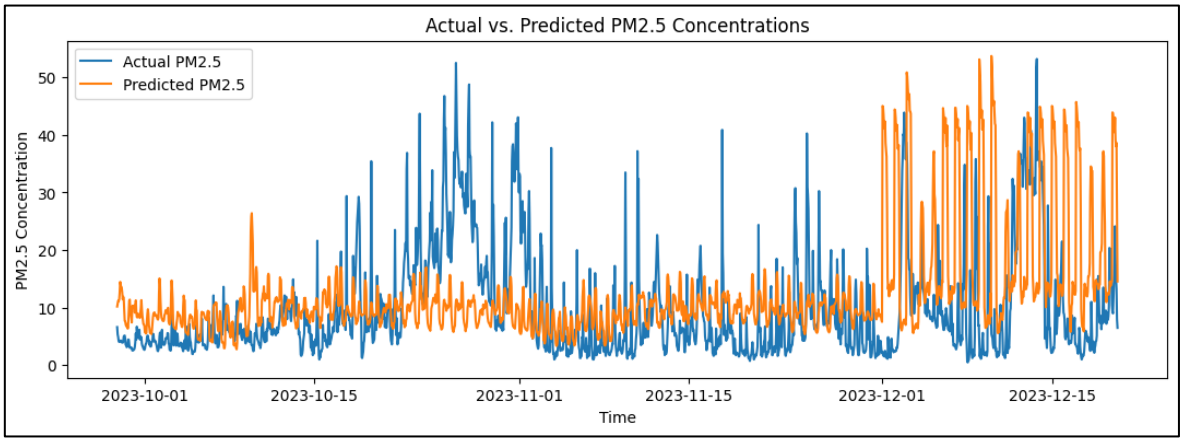


Figure 4.26: Predicted values of Random Forest Model vs Test data values of cluster 2

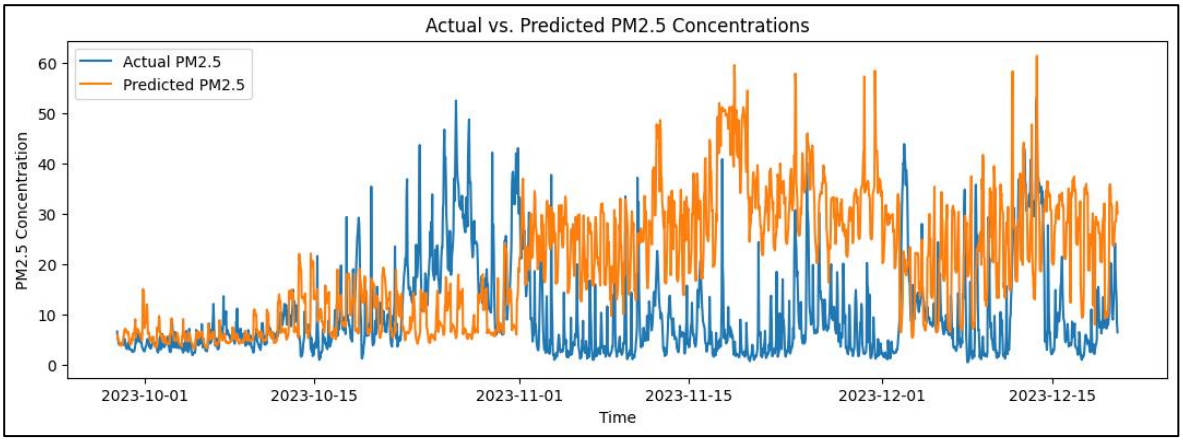


Figure 4.27: Predicted values of Random Forest Model vs Test data values of cluster 3

4.5 Temporal Model Validation

Table 4.1: Temporal Model Validation

Model	Cluster 1		Cluster 2		Cluster 3	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
ARIMA	1.74	2.72	1.87	2.69	2.37	3.96
SARIMA	2.32	3.07	2.75	3.36	3.65	5.43
ETS	5.04	6.12	5.12	6.94	7.58	8.02
LSTM	1.60	2.60	2.08	2.25	1.29	2.60
GRU	1.81	2.65	2.59	2.86	1.45	2.98
RF	8.17	13.64	14.42	18.47	5.71	9.08

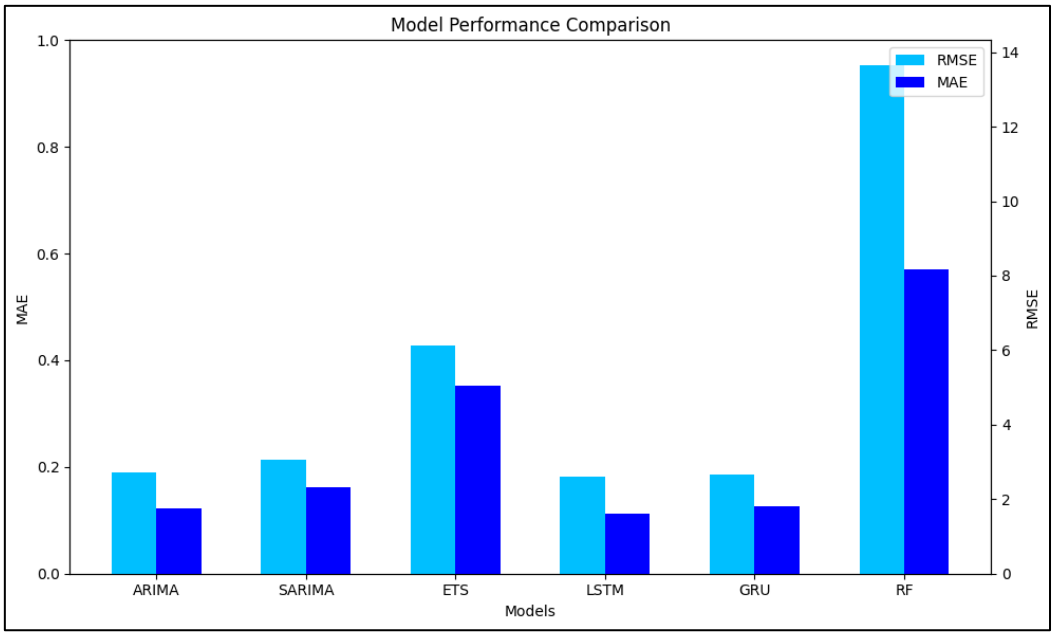


Figure 4.28: Temporal Model Performance comparison

The 4.1 table shows MAE and RMSE values for each model. we have identified LSTM as the most accurate and reliable model for PM2.5 prediction for our dataset. The LSTM model is ideally suited for this situation because of its ability to accurately model long-term dependencies and nonlinear interactions in the data.

4.6 CNN Model Performance Evaluation

4.6.1 Basic CNN Model

In this section, we evaluate the performance of our CNN models by comparing the predicted values generated by the models with the actual test values obtained from the dataset aligned with the same satellite image. The accuracy of our models is compared through visual representations to check how well our models capture the image patterns to estimate variations in PM2.5 levels.

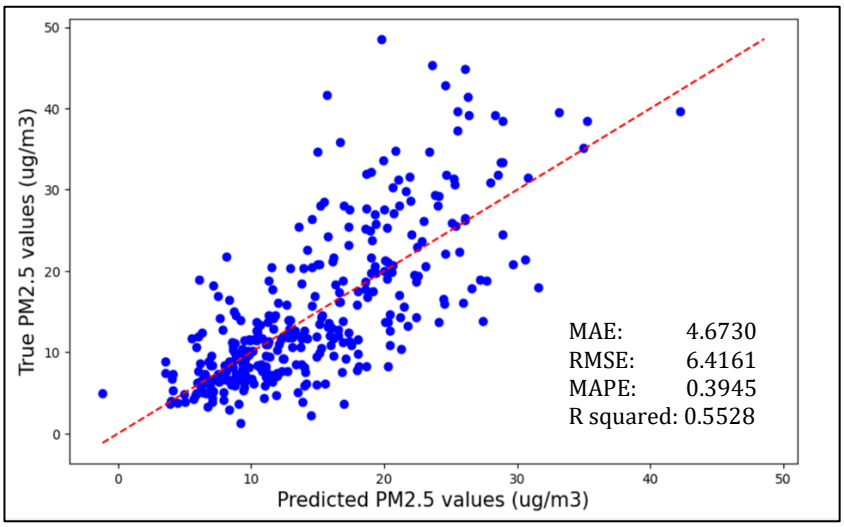


Figure 4.29: Predicted values of Basic CNN Model vs Test data values

4.6.2 Addition of new features to the CNN

Then we tried to introduce new inputs to the CNN to improve the accuracy of CNN. Those features are described in chapter 3.5. We cannot see any improvement in the accuracy of CNN after adding external features.

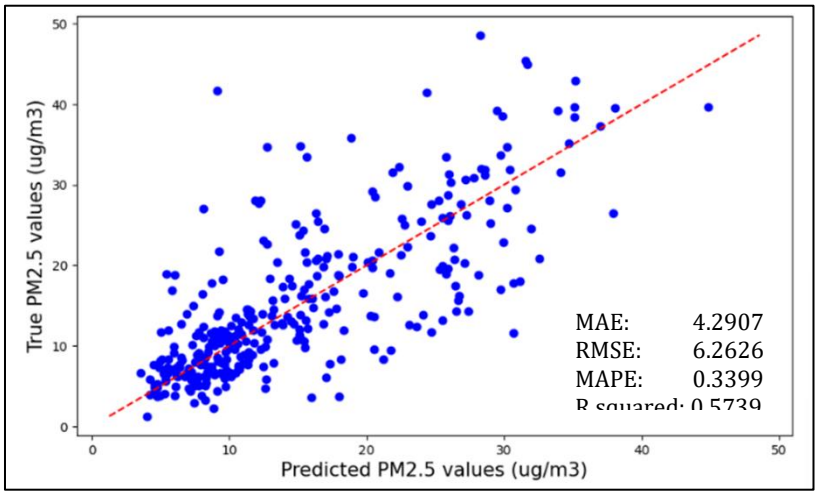


Figure 4.30: Predicted values of CNN Model after adding features vs Test data values

Following is the error variation of the CNN after adding each feature to the CNN. We can see how the error varies after adding the features one by one.

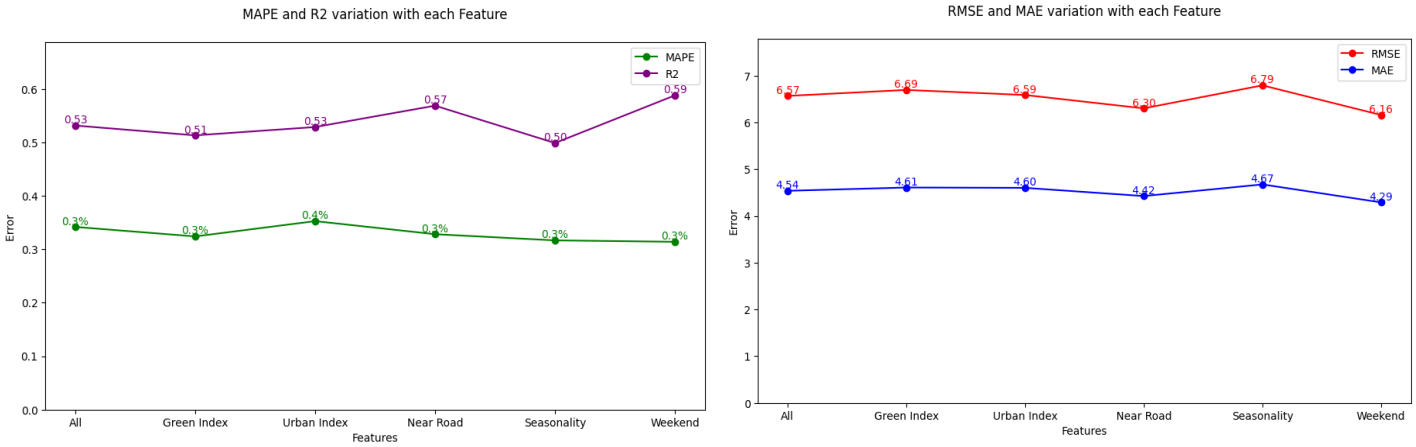


Figure 4.31: Error matrix variation with each feature

4.6.3 Implementation of Transfer Learning

Transfer learning uses pre-trained models for the top layers of a CNN, improving pattern recognition. The following graphs and table show that. Among the three models tested, VGG16 gives the most accurate results by getting a R squared value of 0.6076 while MobileNet gets 0.4733 and Resnet50 gets 0.4533 R squared values. So, we decided to proceed with the VGG16 for the building of the CNN model.

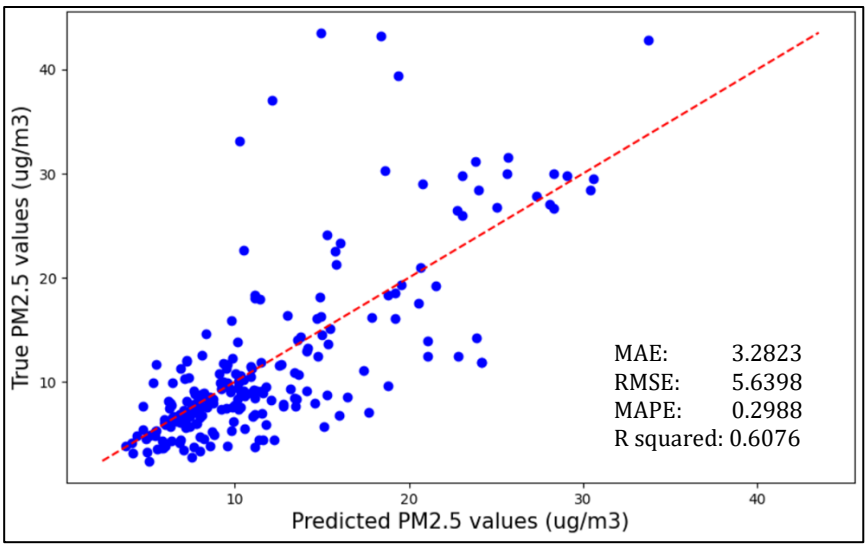


Figure 4.32: Predicted values of CNN Model after using VGG16 vs Test data values

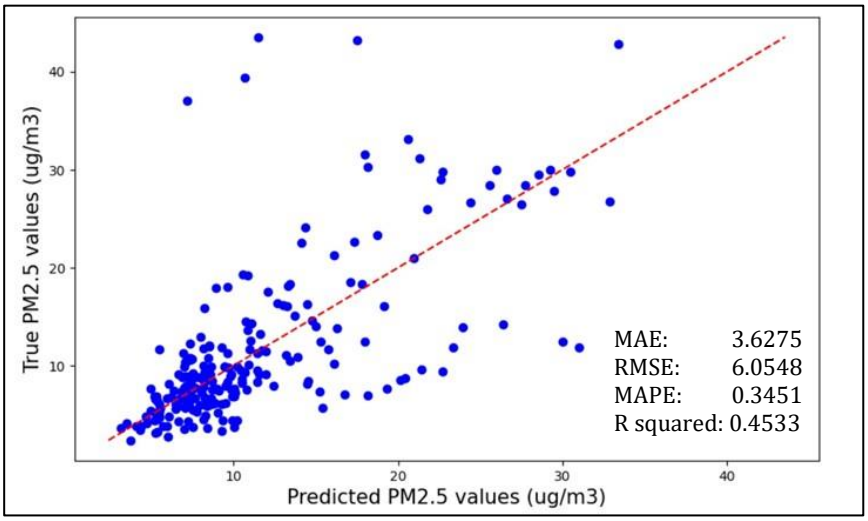


Figure 4.33: Predicted values of CNN Model after using ResNet50 vs Test data values

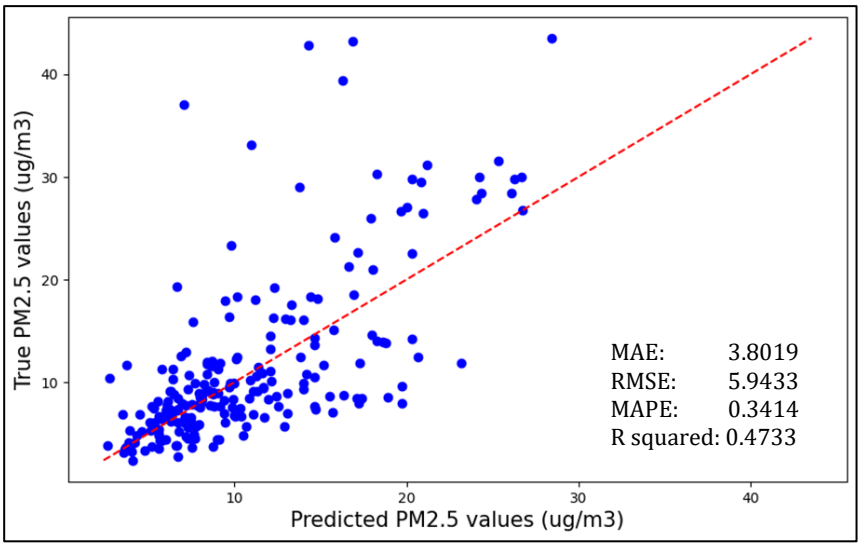


Figure 4.34: Predicted values of CNN Model after using MobileNet vs Test data values

Table 4.2: Transfer Learning Model Validation

Model	MAE	RMSE	MAPE	R squared
VGG16	3.2823	5.6397	0.2988	0.6076
MobileNet	3.8019	5.9433	0.3414	0.4733
ResNet50	3.6275	6.0548	0.3451	0.4533

After doing several steps for optimizing the CNN model we were able to get the accuracy of 84% with the addition of 18 points of lag values from the LSTM model. The final model architecture which yields this accuracy has been described in chapter 3.5.4.

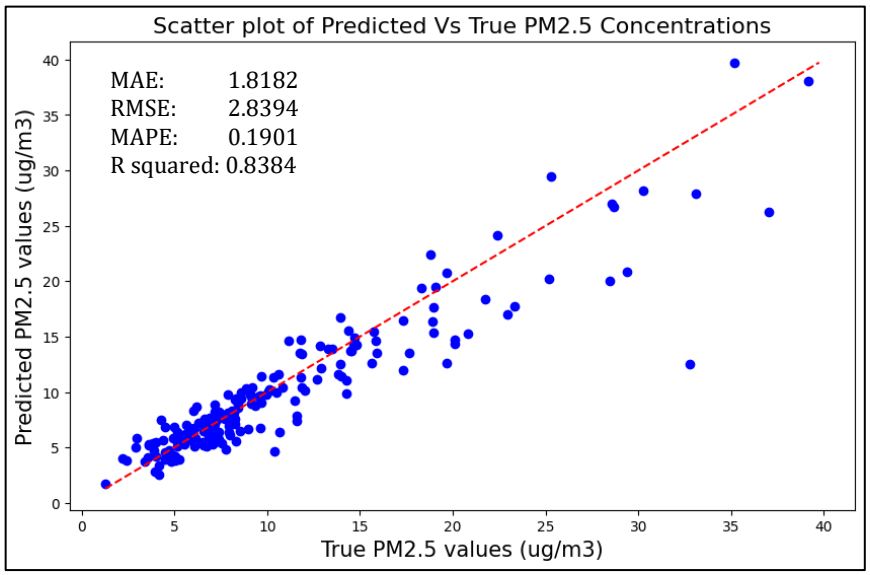


Figure 4.35: Predicted values of Final CNN Model vs Test data values

4.7 Hybrid Model Performance Evaluation

This section reviews the performance of our CNN-LSTM hybrid models by matching the predicted PM2.5 values with the actual test data corresponding to the same satellite images. We compare the accuracy of our models using visual representations to determine how well they capture image patterns and predict PM2.5 level variations.

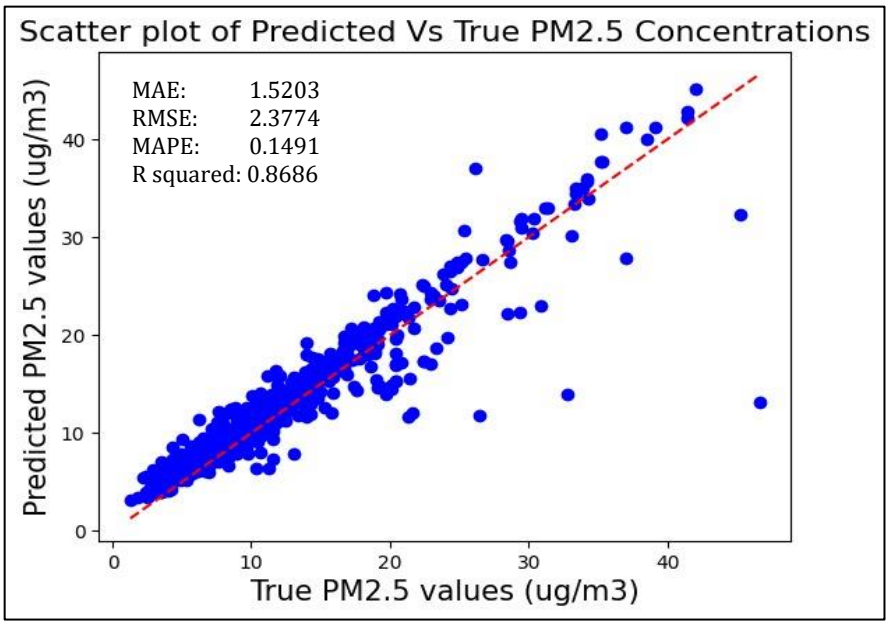


Figure 4.36: Predicted values of Hybrid CNN-LSTM Model vs Test data values

Table 4.3: Model Performance Comparison

	LSTM	CNN	Hybrid CNN-LSTM model
MAE	2.958	1.818	1.520
RMSE	6.215	2.893	2.377
MAPE	0.495	0.190	0.149
R-squared	41.13%	83.84%	86.86%

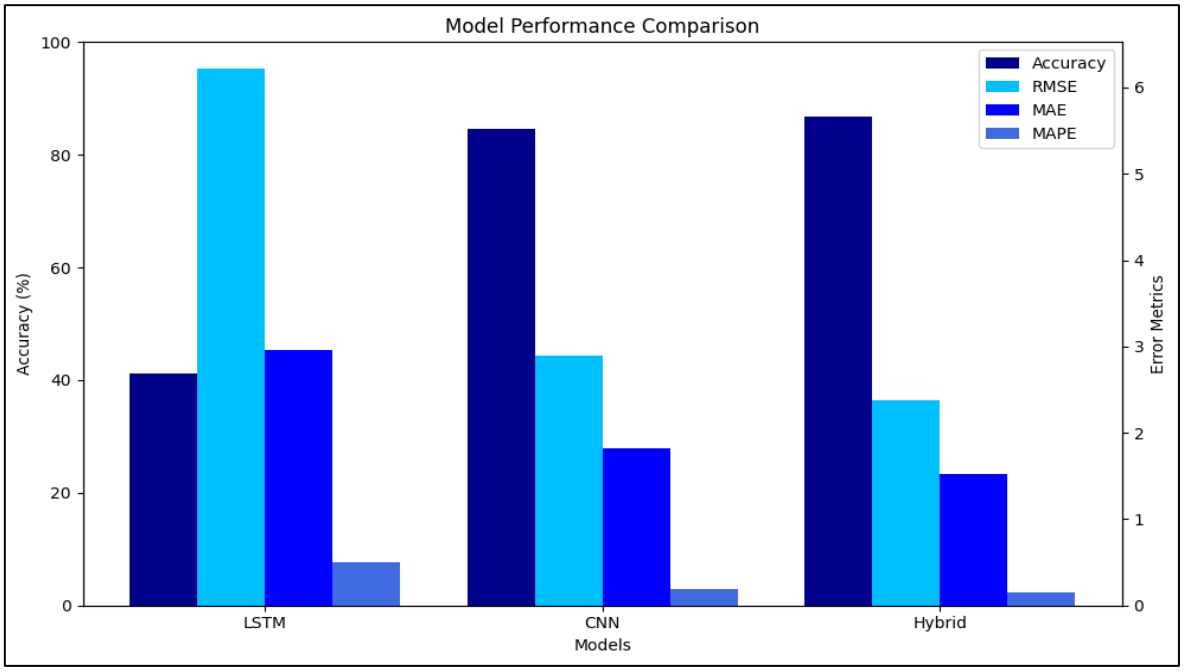


Figure 4.37: Model Performance Comparison

CHAPTER FIVE

CONCLUSION

In this study, different methods for forecasting PM2.5 concentration in 3 clustered were compared. The 4.1 table shows MAE and RMSE values for each model. we have identified LSTM as the most accurate and reliable model for PM2.5 prediction for our dataset. Our results show the significance of using deep learning methods for temporal PM2.5 concentration forecasting, particularly LSTM networks. The LSTM model is ideally suited to handle the dynamic and complex nature of air pollution dynamics in urban locations because of its ability to accurately model long-term dependencies and nonlinear interactions in the data.

The key findings from our project demonstrate that the hybrid CNN-LSTM model significantly improves the accuracy of PM2.5 concentration predictions compared to individual models. The Table 4.2 shows the improvement in accuracy both before and after combining CNN and LSTM. By integrating temporal features from LSTM with spatial patterns captured by CNN, our model effectively predicts air pollution levels in areas lacking sensor networks. The hybrid approach outperformed traditional statistical models like ARIMA, as well as standalone deep learning models such as LSTM and CNN. Our results highlight the model's ability to capture both the temporal dynamics and spatial variability of PM2.5 concentrations, making it a robust tool for air quality forecasting in Sri Lankan urban environments.

The significance of this work lies in its pioneering approach to air quality prediction in Sri Lanka, utilizing deep learning techniques. This research provides critical insights that can inform public health initiatives and policymaking, contributing to the reduction of air pollution in urban areas. A significant challenge encountered during the project was the limited availability of data. With only 6 months of temporal data and 1,422 satellite images, the dataset was insufficient to fully train a deep learning model that typically requires a much larger volume of data to achieve high accuracy. This limitation posed difficulties in capturing the complexities and variations in PM2.5 concentrations across different regions and periods, potentially affecting the model's predictive performance. As a future direction, we recommend the development of a real-time prediction application or website based on our model. This platform could provide users with up-to-date PM2.5 concentration forecasts for various locations, particularly in areas lacking direct sensor networks.

By integrating our hybrid CNN-LSTM model, the application would deliver accurate and timely air quality information, enabling individuals and authorities to take proactive measures to protect public health. This real-time tool could serve as a valuable resource for urban planning, environmental monitoring, and public awareness, further enhancing the practical impact of our research.

This research project has been a challenging yet rewarding journey, offering invaluable insights into the complexities of predicting air pollution using advanced deep-learning techniques. Despite facing significant challenges, particularly the limited availability of data, the project has provided a solid foundation for future work in this domain. Moving forward, we are excited about the potential to expand this work and contribute to the development of real-time prediction tools that could make a meaningful difference in mitigating air pollution's impact on society.

REFERENCES

1. C.-J. Huang and P.-H. Kuo, “A Deep CNN-LSTM Model for Particulate Matter (PM2.5) Forecasting in Smart Cities,” *Sensors*, vol. 18, no. 7, p. 2220, Jul. 2018, doi: 10.3390/s18072220. [1]
2. U. Pak *et al.*, “Deep learning-based PM2.5 prediction considering the spatiotemporal correlations: A case study of Beijing, China,” *Science of The Total Environment*, vol. 699, p. 133561, Jan. 2020, doi: 10.1016/j.scitotenv.2019.07.367. [2]
3. M. Faraji, S. Nadi, O. Ghaffarpasand, S. Homayoni, and K. Downey, “An integrated 3D CNN-GRU deep learning method for short-term prediction of PM2.5 concentration in urban environment,” *Science of The Total Environment*, vol. 834, p. 155324, Aug. 2022, doi: 10.1016/j.scitotenv.2022.155324.[3]
4. T. Zheng, M. H. Bergin, S. Hu, J. Miller, and D. E. Carlson, “Estimating ground-level PM2.5 using micro-satellite images by a convolutional neural network and random forest approach,” *Atmospheric Environment*, vol. 230, p. 117451, Jun. 2020, doi: 10.1016/j.atmosenv.2020.117451.[4]
5. Dhammapala, R., Basnayake, A., Premasiri, S., Chathuranga, L., Mera, K. (2022). PM2.5 in Sri Lanka: Trend Analysis, Low-cost Sensor Correlations and Spatial Distribution. Aerosol Air Qual. Res. 22, 210266. https://doi.org/10.4209/aaqr.210266
6. Multi-directional temporal convolutional artificial neural network for PM2.5 forecasting with missing values: A deep learning approach. https://doi.org/10.1016/j.uclim.2021.100800
7. Educative. (n.d.). Educative Answers - Trusted Answers to Developer Questions. [online] Available at: https://www.educative.io/answers/time-series-prediction-using-lstm
8. Coding ninjas studio, https://www.codingninjas.com/studio/library/time-series-prediction-with-gru
9. Rendyk, “Tuning the hyperparameters and layers of neural network deep learning,” Analytics Vidhya https://www.analyticsvidhya.com/blog/2021/05/tuning-the-hyperparameters-and-layers-of-neural-network-deep-learning/.

10. Khandelwal, Vaibhav. “The Architecture and Implementation of VGG-16.” Medium, 18 Aug. 2020, pub.towardsai.net/the-architecture-and-implementation-of-vgg-16-b050e5a5920b.
11. Mukherjee, Suvaditya. “The Annotated ResNet-50.” Medium, 18 Aug. 2022, towardsdatascience.com/the-annotated-resnet-50-a6c536034758.
12. Polukhin, Andrii. “MobileNet Architectures.” Medium, 17 Oct. 2022, medium.com/@pandrii000/mobilenet-architectures-17fe7406d794.