Air Pollutant Forecasting using Deep Learning

By

Ketul Dave

A thesis submitted in partial fulfillment

of the requirements for the degree of

Master of Science (MSc) in Computational Sciences

The Faculty of Graduate Studies

Laurentian University

Sudbury, Ontario, Canada

THESIS DEFENCE COMMITTEE/COMITÉ DE SOUTENANCE DE THÈSE
Laurentian Université/Université Laurentienne
Office of Graduate Studies/Bureau des études supérieures

Title of Thesis Titre de la thèse	Air Pollutant Forecasting using Deep Learning	
Name of Candidate Nom du candidat	Dave, Ketul	
Degree Diplôme	Master of Science	
Department/Program Département/Programme	Computational Sciences	Date of Defence Date de la soutenance September 16, 2021

APPROVED/APPROUVÉ

Thesis Examiners/Examinateurs de thèse:

Dr. Kalpdrum Passi
(Supervisor/Directeur(trice) de thèse)

Dr. Ratvinder Grewal
(Committee member/Membre du comité)

Dr. Oumar Gueye
(Committee member/Membre du comité)

Dr. Pradeep Atray
(External Examiner/Examinateur externe)

Approved for the Office of Graduate Studies
Approuvé pour le Bureau des études supérieures
Tammy Eger, PhD
Vice-President Research (Office of Graduate Studies)
Vice-rectrice à la recherche (Bureau des études supérieures)
Laurentian University / Université Laurentienne

ACCESSIBILITY CLAUSE AND PERMISSION TO USE

I, **Ketul Dave**, hereby grant to Laurentian University and/or its agents the non-exclusive license to archive and make accessible my thesis, dissertation, or project report in whole or in part in all forms of media, now or for the duration of my copyright ownership. I retain all other ownership rights to the copyright of the thesis, dissertation or project report. I also reserve the right to use in future works (such as articles or books) all or part of this thesis, dissertation, or project report. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that this copy is being made available in this form by the authority of the copyright owner solely for the purpose of private study and research and may not be copied or reproduced except as permitted by the copyright laws without written authority from the copyright owner.

Abstract

In nearly every country, air pollution has become a serious issue. whether it is developing countries or developed countries, as urbanization and industrialization has increased. Governments and citizens are greatly concerned about air pollution, which has a negative impact on human health, the well-being of all life forms, and global economic development. Numerical data is used in traditional air quality forecast systems, which necessitates more computing resources for pollutant concentration measurement and yields poor results. We used a commonly used deep learning model to solve this problem. Particulate Matter 10 was the pollutant studied in this study (PM10).

This study examines the methods and techniques for predicting air quality using Deep Learning. Various deep learning models have been investigated. This research incorporates a recurrent neural network (RNN), a long short-term memory (LSTM), a gated recurrent unit (GRU) and a bidirectional long short-term memory combination for forecasting. The dataset is primarily comprised of pollution and meteorological time series data from AirNet China and the United States Environmental Protection Agency. We studied various architectures and their variations in topologies and model parameters in order to decide the best architecture. The Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE) were used to assess the models (MAPE). Each experiment was run for up to 1000 epochs by varying the learning rate, the number of nodes in a layer, and the total number of hidden layers. All models performed admirably in terms of prediction, according to the results. For AirNet dataset GRU based architecture produced best outcome while for EPA dataset LSTM based architecture outperformed other models.

Keywords*:* PM10, RNN, LSTM, GRU, MAPE, RMSE, BiLSTM

Acknowledgments

First and foremost, I am thankful to God for my good health and well-being, which enabled me to graduate and finish this thesis.

Prof. Kalpdrum Passi has been an excellent instructor, mentor, and thesis supervisor, providing insightful and humorous advice and support. My experience working with him is something I am proud of, and I am grateful for it. He was always available to clear all my doubts and handled all the queries with patience. He helped me a long way since I started my academic journey with Laurentian.

In addition, I would want to express my gratitude to my parents for their sound advice and sympathetic ear. You are always inclined to assist me. They helped me to achieve my dream goal by providing mental as well as financial support throughout this roller coaster ride.

Finally, I could not have finished this dissertation without the help of my friends, who provided fascinating talks as well as enjoyable distractions from my study.

Table of Contents

List of Tables

List of Figures

LIST OF ABBREVIATIONS

AI	Artificial intelligence
ANN	Artificial Neural Network
NNBP	Neural Networks backpropagation
ANFIS	Adaptive neuro-fuzzy inference system
NARX	Nonlinear autoregressive exogenous
FS	Feature selection
GA	Genetic algorithm
RMSE	Root Mean Squared Error
MSE	Mean Squared Error
AI	Artificial Intelligence
API	AQI Value of The Next Day
AQI	Air Quality Index

POLLUTANTS	PM2.5	$\mu g/m^3$	Daily averaged concentration of PM2.5
	PM10	$\mu g/m^3$	Daily averaged concentration of PM10
	O_3	$\mu g/m^3$	Daily averaged concentration of O3
	NO_2	$\mu g/m^3$	Daily averaged concentration of NO2
	SO_2	$\mu g/m^3$ or ppb	Daily averaged concentration of SO2
	CO	$\mu g/m^3$	Daily averaged concentration of CO
	T	Fahrenheit	Temperature

	P	mig/m^3	Pressure
	RH	%	Relative Humidity
Meteorological Data	UwC	m/s or Knots	U wind Component (positive for a west to east flow / eastward wind)
	VwC	m/s or Knots	V wind Component (positive for south to north flow / northward wind)
	PR	Mm	Precipitation Rate
	CC	Okta	Total Cloud Cover

CHAPTER 1

Introduction

1.1 Background

Air pollution refers to the contamination of the air, whether indoors or outdoors. When contaminants enter the environment, the air becomes polluted, making it impossible for plants, animals, and humans to endure [1]. The atmosphere, which is made up of a variety of gases is responsible for the survival of all living beings. Changes in these gases can create an imbalance that can be dangerous to survival. Pollution is a major issue in our society because it impacts humans, wildlife, the environment, and other living things. Pollution can cause unpredictable climate change, wreaking havoc on the atmosphere and harming animals and other living beings. In recent years, air pollution has reached new highs in almost every country on the planet. The amount of dust in the air is increasing, and the environmental consequences of urban transformation processes are being ignored [2]. One of the leading causes of death in the world is air pollution. Pollution was responsible for 6.6 million deaths in 2020. Furthermore, pollution was responsible for 19% of all cardiovascular deaths, 21% of stroke deaths, and 23% of lung cancer deaths around the world (World Health Organization: https://www.who.int/). As a consequence, we need a reliable forecasting approach to direct us to a critical position in disaster management and emergency preparedness [3].

Urbanization is one of the primary causes of air pollution since increased transportation facilities emit more pollutants into the atmosphere and industrialization is another major driver of

air pollution. Nitrogen oxide (NO), carbon monoxide (CO), particulate matter (PM), Sulphur dioxide (SO2), and other pollutants are the most common. Carbon monoxide is created when a propellant, such as petroleum or gas, is not properly oxidized. When thermal fuel is ignited, nitrogen oxide is created. Carbon monoxide causes headaches and vomiting; benzene, which is formed because of smoking, causes respiratory issues; and nitrogen oxides induces dizziness and nausea. Particulate matter having a diameter of less than 2.5 micrometers has a greater impact on human health. Environmental measures must be made to reduce air pollution. The Air Quality Index (AQI) is a criterion for analyzing air quality. Previously, traditional approaches such as probability and statistics were employed to estimate air quality, but these approaches are extremely difficult to employ to forecast air quality. Because of technological advancements, it is now relatively simple to get data regarding air contaminants using sensors.

The evaluation of raw data in order to discover contaminants necessitates a thorough examination. Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Deep Neural Networks (DNN), and machine learning algorithms ensure that future AQI may be predicted and necessary precautions can be implemented. Supervised Learning, Unsupervised Learning, and Reinforcement Learning are three types of learning algorithms used in machine learning, which falls under artificial intelligence. We adopted a supervised learning strategy in the proposed research.

Governments, Environmental administrators and people all over the world are becoming deeply concerned about air quality. New technologies are constantly being developed to increase air quality awareness around the world. Air quality mapping is improving all the time, thanks to smart city advances and the growing number of internet-of-things sensor devices. The Increased data production aids in the acceleration of air pollution activity. Air pollution forecasting, or the

estimation of the atmospheric composition of pollutants for a given time and place is a popular research subject. With a reliable air quality forecast, one can plan ahead of time. Because of the health effects of air pollution, people can choose how to behave. Boznar et al. [6] conducted the first research for forecasting and modelling air pollutants. Since its introduction in 1993, the Artificial Neural Network (ANN) has been regarded as the most important technique in this field. The majority of the input parameters for the built ANN network model came from current scientific literature, arbitrary inference, and common sense [4]. Some researchers used a genetic algorithm optimization operation for input variable selection prior to ANN training and compared the forecasted outcomes to various linear regression models [5]. The goal of this research is to create an artificial neural network that can forecast pollution levels in the air.. Air quality forecasting can help with the study of changing patterns in different types of emissions. It will also help with safety measures preparation and management. This study leads to hyper-parameter-tuning based model training to clear excrescent information, reduce multi-collinearity, and improve generalization efficiency.

1.2 Air Pollution

Humans in the twenty-first century face new threats as society develops and improves in each sector. The technological revolution, population growth, Transportation, Deforestation, gradual climate change, and a slew of other problems placed the world and atmosphere in jeopardy. Air pollution is one of these issues that has piqued the interest of countries, the United Nations, and scholars/researchers all over the world. Globalization opened up new perspectives to the world. Essentially, the planet has become a shared home for people, with both negative and positive deeds affecting all in this century which increased the pace of technological revolution and increased the pace of Air Pollution. Air pollution is caused by a large and unhealthy volume

of various pollutants, such as gases, particulates, and biological molecules [7]. These hazardous substances are produced, combined, and released into the atmosphere. It causes illnesses, allergies, and the deaths of millions around the world. Due to the fact that air pollution is causing more deaths around the world, governments, people and everyone else who can help reduce the exponentially rising pollutants bear responsibility for combating these phenomena. Dealing with this problem of Air Pollution requires additional guidance and research due to the complexity of the topic. As the cause of air emissions varies from region to region or nation to country, many chemical elements contribute to air pollution. These components must be discussed if we are to combat air pollution; however, they must be addressed separately due to their properties and requirements. To combat these common threats, collective interest and disadvantage necessitate cooperation and shared strategies.

1.2.1 Source of Air pollution

The sources of air pollution can be categorized into two parts, man-made and natural sources. The natural sources of air pollution do not count much for the current increase in air pollution as compared to man-made sources. The major sources of pollution are listed below.

1. The Burning of Fossil Fuels
2. Industrial Emission
3. Indoor Air Pollution
4. Wildfires
5. Microbial Decaying Process
6. Transportation
7. Open Burning of Garbage Waste
8. Construction and Demolition

9. Agricultural Activities

10. Use of chemical and synthetic products

1.2.2 Effects of Air pollution

Air pollution affects the earth in various ways, from health to agriculture and economics. Each of them should be addressed separately.

1.2.2.1 Health Effects

The introduction of the chemicals mentioned above into our environment poses a threat to our health. Many illnesses are caused by toxins and certain people also die as a result of it. Air exposure affects a variety of respiratory issues, including difficulty breathing, wheezing, coughing, asthma, and heart complications. These factors have an effect on the human body and the body environment in general [8]. The following is a list of the health consequences of air pollution, to be more specific.

• Deaths

• Cardiovascular disease

• Lung disease

• Lung cancer

• Infants

• Central nervous system

1.2.2.2 Agriculture and economic effects

Agriculture and other economic conditions are impacted by this global threat. Crop yields in India's most contaminated regions have decreased by half, according to the study [9]. Meanwhile, according to a report conducted by the World Bank and the University of

Washington's Institute for Health Metrics and Evaluation (IHME), air pollution costs the global economy $5 trillion per year due to lost productivity and quality of life [10].

1.3 Objectives

To forecast air pollution levels, it is critical to manage and use historical data and calculate parameters while taking into account the availability of large volumes of data in order to distinguish the type and degree of relationship for appropriate and effective information extraction. Many spatial data are unstructured in nature, such as air quality, which has varying levels of pollution at various places. For all intents and purposes, the method of obtaining a continuous data set from a sparse data repository is useful. This study aims to forecast air quality using historical hourly and daily data from China and California, USA, using artificial intelligence approaches such as artificial neural networks. In essence, putting algorithms in place will result in the generation of performance results based on data prediction. More detailed findings and precise and reliable evidence can be used to categorize comparable data. Accurate and relevant data will help in the implementation of effective environmental and climate policies.

Exact air pollution forecasting is tedious and critical, which may be useful in fields such as ozone layer depletion and global warming forecasting. Formulating and making air quality forecast calculations that are based on similarities will have accurate and dependable performance forecasts. Inaccurate forecasting wastes energy and time and it can lead to ineffective control crises such as poor air quality, which can affect people and result in poor pollution management. There is a need to develop a good air pollution forecasting system and more importantly, a system that can be more reliable and precise than existing air pollution forecasting systems.

This study is based on historical hourly data from April 2015 to July 2017 for China and daily data for California USA from 1980 to 2020. The data consists of both types of features which

include meteorological and pollute concentration features for AQI during the observed period. Pollutant Concentrate data included particulate matter PM10, PM2.5, Nitrogen Dioxide (NO2), Sulphur Dioxide (SO2), Carbon Mono-Oxide (CO) and ozone (O3). Meteorological data includes Temperature, Pressure, Relative-Humidity, U wind Component, V wind Component, Precipitation Rate, and Total Cloud Cover.

The objectives of this study are to use machine learning and deep learning algorithms to improve the predictability of the AQI.

1.4 Outline of the Thesis

This study has the following organization.

Chapter 1 includes a background on air pollution, an overview of the research, and a discussion of the problem statement.

Chapter 2 reviews previous research on forecasting of air pollution.

Chapter 3 presents the air pollution data, pre-processing of data and modelling.

Chapter 4 highlights the analysis of data and discusses the results obtained by different models. It discusses the Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE) to assess the result that we analyzed for accurate air pollution forecasting.

Chapter 5 gives the conclusions and directions for future work.

CHAPTER 2

Literature Review

Human activities and climate change have a negative effect on the environment and air quality in our industrialised world. The air contains a large number of chemicals, particulates, and molecules as a result of the industrial revolution and massive human activity. As a result, these components contribute to air pollution, which causes a wide range of illnesses, including asthma, and even the death of millions of people around the world. According to a survey conducted by the World Health Organization in 2014, it is estimated to kill 7.5 million people worldwide. (https://www.who.int)

Following the technological age, complex technology and supercomputers, as well as advances in algorithms, made it possible to use these technologies to forecast future air emissions in a given geographic region based on historical data on air pollution in that area. One of these rapidly evolving technologies is machine learning, or artificial intelligence in general. As a result, researchers have been working on improving machine learning algorithms for air quality forecasting. It tries to clarify the indicators of what factors affect air quality and propose models to predict the air quality using previously stored data of a given region. These algorithms are extremely effective in that they recognise the indication of a transition in this regard.

As air pollution is one of the most serious issues in cities, especially metropolitan cities, researchers attempted to focus on each of them separately [11]. There are several different compounds and densities. As a result, the analysis would be narrowed down to a specific area or

regions in order to be more accurate and useful. Neural networks, on the other hand, would take precedence in terms of the mathematical component, as the study's algorithms are neural networks.

To model time-series PM2.5 concentrations in Japan's air, Theang et al. [12] used a dynamically pre-trained Deep Recurrent Neural Network (DRNN). In their proposed pre-training scheme, the weights of the networks shift over time to reach a dynamically and sequentially increasing outcome, resulting in a more accurate learning representation of time-related input data. They used data from physical sensors to observe the air quality. Spatial consistency in the physical orientation of the selected sensors was taken into account to increase the DRNN prediction accuracy. While collecting sensor data, they took into account PM2.5 concentrations, wind speed and location, temperature, illuminance, humidity, and rain. They've also proposed a strategy for decreasing computing expenses by removing sensors that don't make a significant contribution to better forecasts. For sparsity-based sensor filtering, they used the elastic net method. DRNN has proved to be more reliable than the auto-encoder training form. DRNN, on the other hand, hasn't thought about how to deal with anomalous sensor data in the context of prediction.

Kurt et al. [13] demonstrated an online air quality forecasting method. The focal point is Greater Istanbul. Nikov et al. [33] developed a website AirPolTool (no longer active) [34] that publishes Istanbul's air pollution for the next three days, with data updated twice a day for more accurate results. The researchers used a neural network to forecast the amounts of three air pollutant indices for the next three days, including SO2, PM10, and CO. According to the research, a simple neural network can predict the amount of air pollutant indicators with high accuracy. Simultaneously, the research uses several optimization techniques, such as multiple input parameters to help improve the results. Their training method is straightforward: it forecasts the next day based on the previous day's data and the merged results for the other days in the same

sequence. The best set of historical results comes from several checks, which takes 3-15 days. Then the “day of the week” was examined as an input parameter to see whether it influenced accuracy. The results show that it helps the machine make more accurate predictions.

Wan & Lei [14] found that air pollution in Macau, China is in tandem with economic growth. Due to the harms and health impacts, it causes, monitoring and predicting the Air Pollution Index (API) became increasingly important for city residents. Their study proposes an adaptive neuro-fuzzy based method for predicting API in that city. Sulphur dioxide and total suspended particulates from past and historical records, as well as other variables and concentrations, are used in the model's input and prediction mission. In this study, backpropagation with the least square algorithm was used as a learning process. For preparation and capacity improvement, they used ten years of historical data from Macau. The studies, according to the report, indicate a satisfactory success in predicting air quality in Macau.

Cai & Xie [15] present an artificial neural network approach for predicting air pollution concentrations in Guangzhou city of China for an hourly location. Outside of cities, a variety of variables influence the air pollution levels, but they all have a complex and intertwined relationship. There are four categories for traffic, background concentrations, meteorological, and geographical factors. In the ANN-suggested strategy, many of these classified variables are used as inputs. Pollutant concentrations of CO, NO_2, PM_{10}, and O_3 were expected. These pollutants were found to be the most important sources of emissions in the air. Data was collected from two locations along the arterial (location in Guangzhou, China) using self-driving car units. The aim of the thesis is to predict average hourly concentrations in the cities described. The forecast is only valid for a period of 10 hours. The findings show that the back-propagation neural network is capable of accurately predicting the four pollutant elements mentioned. The model is frequently

linked to other multiple linear regression models and the California line source model, with value estimation evaluation techniques like MRE, MAE, RMSE, and Correlation coefficient being used in the study. Experiments indicate that the suggested technique performs better than existing models in predicting concentrations properly. The experiments show that the presented approach outperform those models and predict concentrations more accurately.

In Santiago, Chile, Pérez & Reyes [16] conducted an experiment to estimate average PM2.5 concentrations hourly. They focus their efforts in areas where PM2.5 levels are the highest. The concentration hits an all-time high from May to September. This month, data for the years 1994 and 1995 was collected. According to the paper, by fitting a function to measure the previous day's 24 hours and taking into account the variations that will occur, they can forecast the day's concentrations. The data used for planning and evaluation for both years is from May 1 to September 30. Throughout the year, 24 datatypes with a total of 25 columns were created. To see if the prediction task is successful, three models were tested: multilayer neural networks, linear regression, and persistence. The neural network model outperformed the other models in general. The level of consistency, in reality, varies depending on the time of day. Prediction errors were 30 percent early in the day, but by late in the day, they had risen to 60 percent. The study's goal was to find out the reasons for poor accuracy in prediction. For instance, noise, jumbled data, and so on.

Another research was performed by Kolehmainen & Ruuskanen [17] to forecast the hourly average of NO_2 and basic meteorological variables. Data was collected in Stockholm from 1994 to 1998. In order to assess their ability and probability of making this prediction, the study looked at two simple and distinct neural network models. SOM and MLP have been tested in a number of ways, including using periodic components, applying neural network techniques to residual values

without periodic components, and only using ANN. The final result shows that using raw data with the MLP network produces the best results.

Shi et al. [18] proposed using a spatial temporal model with RNN and convolutional LSTM to predict precipitation over the next two hours. Air quality estimation is similar to climate estimation; however, two parameters make air quality conjecture more complex and distinct from predicting precipitation: 1) Air quality forecasting takes longer than climate forecasting; the earlier often forecasts in 4 to 5 days, and even 10 days on rare occasions. 2) when assessing air quality, additional persuasive factors such as the progression of atmospheric poisons and their relationship to meteorological conditions must be considered.

Liang et al. [19] released a dataset containing PM2.5 estimations that were recently calculated in Beijing. Following that, Liang et al. [20] disseminated a larger dataset to dissect the poison element in five Chinese cities. Many of the datasets used above are point-by-point files, which makes it difficult to illustrate in a spatially explicit manner.

Due to the high nonlinear processes that involve pollution concentrations and their partly understood dynamics, developing a model capable of anticipating these types of occurrences is exceedingly challenging [21]. ML models are nonparametric and nonlinear models that discover the underlying link between variables using only historical knowledge [22]. When forecasting time series (TS) with a high level of nonlinearity, ML techniques such as artificial neural networks (ANNs), genetic programming (GP), and support vector machines (SVMs) have been found to beat ARIMA. Sharda and Patil [23], for example, compared the outcomes of an ANN with ARIMA.

Daz-Robles et al. [24] used an empirical investigation to forecast air quality in Chile, specifically PM10 data, using a hybrid model combining ANNs and ARIMA. The ARIMA model was used to capture the dataset's linearity, while ANNs were used to capture the nonlinearity found in the ARIMA model residuals. The authors determined that the resultant model has a high degree of generalisation and outperforms both ARIMA and ANNs when employed separately.

When forecasting hourly air pollution concentrations, Cai et al. [25] compared the results produced using a multilinear regression model to those obtained using an ANN, indicating that ANNs generate more robust results. GP was used by Pires et al. [26] to predict daily averages of PM10 concentrations, and it was compared to partial least square regression (PLSR). Tikhe [27] used both ANNs and GP to forecast air quality in India. When it came to estimating air pollution concentrations, both systems performed admirably, but when it came to short-term forecasting, GP outperformed the others. Castelli et al. [28] proposed an evolutionary method based on other pollution concentrations to forecast ozone concentrations one hour ahead with GP. The method produced accurate findings, exceeding state-of-the-art machine learning approaches.

Many papers have been published that employ support vector machines (SVMs) to forecast time series, and various authors have used SVM to develop models to anticipate air quality and pollution levels. In particular, Drucker et al. [29] presented a support vector regression (SVR) form of SVM for use in regression issues, which may be particularly suitable for this sort of assignment. Müller et al. [30] did research in which SVR was compared to ANNs. Overall, the scientists concluded that SVR performed better. Cao [31] proposed a hybrid time-series forecasting strategy that used ANNs to split the input space and SVMs to model each portioned region. The findings revealed that this hybrid strategy delivers excellent prediction accuracy while also allowing for efficient learning. SVMs were used by Wang et al. [32] to anticipate daily ambient air pollution.

Kurt and Oktay [35] used spatial models with neural networks for forecasting air pollution indicator levels. The pollutants SO_2, CO, and PM10 are the ones that cause the most pollution in cities and are the most difficult to predict. Neural networks were used to create several spatial models for air pollution forecasting. The accuracy of these models in predicting was investigated through a series of tests for an Istanbul district. Geographic models with spatial characteristics outperformed non-geographic models, according to the findings. The three-site distance-based model, in particular, beat all other models by a substantial margin. The results also showed that for each projected day, the best neighbouring district or districts producing the lowest error may be found experimentally. A real-time forecasting system can use the air pollution values from these districts. The concepts shown here are easily adaptable to different parts of the world.

Prachi et al. [36] discovered that the neural network approach is considered as a reliable and cost-effective solution for the task of prediction in another research. Neural network models might be enhanced in an operational context for every given location through testing and fine-tuning. Additional variables (e.g., more weather components, synoptic patterns, traffic flow, or day-of-the-week information) or altering the nature of the model are likely methods (e.g., changing the number of hidden nodes and layers in neural networks). The use of fuzzy modelling, both alone and in conjunction with ANN, is further demonstrating the positive aspects of soft-computing applications in air pollution modelling.

Predictive models of environmental contaminants must be properly tailored according to the climatic and topographical parameters of the city in question, as well as the data availability for the region in question, according to research by Cortina-Januchs et al. [37]. Because of the limitations in the data and variables recorded by permanent monitoring stations, prediction using Artificial Neural Networks was found to be more suited for their case study. They investigated the

advantages of using a clustering method in predicting to find data about the connections between PM_{10} concentrations and meteorological factors (wind speed, wind direction, temperature, and relative humidity). This data was used in an MLP to forecast the average PM_{10} concentration over the following 24 hours by combining it with other time–windows. The connections between meteorological factors and air contaminants were illustrated using clustering techniques, which went beyond human vision. These algorithms can supplement the ANN's input patterns by recognising groupings of data with similar features and discovering correlations between them that would otherwise be unavailable. These connections enable the ANN to generate more accurate predictions. The findings demonstrate that ANNs coupled with clustering methods have greater generalisation capabilities than ANNs and multiple linear regression alone.

The researcher has used different types of architectures to combat time-series types of datasets. Theang [12] tried to use DRNN based architecture to predict PM2.5. Due to the non-linear type of dataset, DRNN was unable to produce accurate results. Cai & Xie[15] used ANN-based architecture and were able to achieve quite good numbers in terms of RMSE and MAPE. Another research [54] has shown that deep learning methods can be combined to predict time-series datasets. Non-linear data prediction can be improved by combining modern deep learning algorithms and tuning the hyperparameters.

CHAPTER 3

Material and Methods

3.1 Types of air pollutants

The Air Pollutants may be comprised of solid particles, gases, or liquid droplets. The pollutant can be categorized as primary and secondary pollutants. We have considered the Primary Air Pollutants in our research. Table 3.1 lists all the air pollutants with their properties.

Table 3.1 Types of air pollutants

Type of Pollutant	**Pollutant**	**Details**
Primary pollutants	**Carbon dioxide (CO_2)**	• CO_2, has been described as the leading pollutant. • It is produced by almost all the human activities. • The combustion of fossil fuels emits billions of metric tons of CO2 each year.
	Sulphur Oxides (SOx)	• SO_2 is likely produced in many industrial processes. • It is also found in coal and petroleum industries.
	Nitrogen Oxides (NO_2)	• NO_2 primarily gets in the air from the

		burning of fuel. • NO_2 produced by off-road and on-road automobiles such as, trucks, cars, buses, and power plants.
	Carbon monoxide (CO)	• Cars, trucks, and other fossil-fuel-burning engines and equipment are the most significant causes of CO in the outdoor air. • CO is released from a number of household products, including unvented kerosene and gas space heaters, leaked chimneys and furnaces, and gas stoves, both of which may have an effect on indoor air quality.
	Volatile Organic Compounds (VOC)	• VOCs are a common form of air pollution. • These emissions are divided into two categories: methane and non-methane. According to researchers, methane (CH_4) is a very effective gas for reducing global warming. • The non-methane volatile organic compounds (NMVOCs) benzene,

		toluene, and xylene are potential cancer-causing agents and can cause leukemia.
	Particulates Matter (PM)	• Particulate Matters are described as small solid or liquid particles suspended in a gas. • Aerosol is a mixture of particles and smoke. • Furthermore, some human operations, such as burning fossil fuels, power stations, and manufacturing processes, produce a lot of aerosols. • These pollutants accumulate quickly and cause health problems such as heart disease, lung function, and cancer [38].
	Others	• Other toxins have an impact on the air and contribute to the degradation of air quality. Persistent free radicals, poisonous, ammonia, metals, chlorofluorocarbon, odors, and radioactive contaminants are among

		these pollutants.

3.2 Data Description

3.2.1 AirNet Dataset

In this research, we used an AirNet Dataset [39], to examine air quality, which includes 6 air quality indices from 1498 monitoring sites. The dataset is at least 40 times larger than most previous datasets.

The data comes from the China National Environmental Monitoring Center (CNEMC) [39], which regulate 1498 monitor stations around the country. Every station continuously monitors air quality and reports the concentrations of various air contaminants every hour. Additionally, the authors of this dataset collected meteorological data from the Global Forecast System (GFS), which includes the six meteorological condition features listed in Table 3.2.

Every 6 hours, the National Oceanic and Atmospheric Administration (NOAA) releases a 3-dimensional matrix to GFS. To acquire AirNet data, this three-dimensional matrix is interpolated with two-dimensional data to generate a four-dimensional matrix. The four-dimensional database includes data on (latitude, longitude, time steps, features). The information was gathered between April 1, 2015, and September 1, 2017. There are six GFS features and seven air quality indicators for each time frame. The dimension of the data is (132,228,707,213).

Table 3.2 GFS Field Description

Feature No.	Name	Description	Unit
001	tmp	Temperature	[K]
002	rh	Relative Humidity	[%]
003	ugrd	U-Component of Wind	[m/s]
004	vgrd	V-Component of Wind	[m/s]
005	prate	Precipitation Rate	[kg/m2/s]
006	tcdc	Total Cloud Cover	[%]

3.2.2 EPA USA Dataset

The AQS Data Mart (EPA USA) [40] is a database that contains all of the data from Air Quality Systems. It includes all of the information acquired by the Environmental Protection Agency (EPA) as part of the national ambient air monitoring network. It also includes the EPA's aggregate numbers (8-hour, daily, annual, etc.). The AQS Data Mart is a web-based tool that makes a weekly copy of AQS available to the public. The Data Mart is aimed at air quality data analysts in the regulatory, academic, and health research sectors. It is designed for people who need to obtain huge amounts of comprehensive technical data from the Environmental Protection Agency (EPA), but it does not include any interactive analysis capabilities.

The most frequently requested data aggregation levels (and key metrics in each) are:

- Sample Values (2.4 billion values back as far as 1957, national consistency begins in 1980, data for 500 substances routinely collected)
- The sample value converted to standard units of measure (generally 1-hour averages as reported to EPA, sometimes 24-hour averages)

- Local Standard Time (LST) and GMT timestamps
- Measurement method
- Measurement uncertainty (where known)
- Any exceptional events affecting the data
- National Ambient Air Quality Standards (NAAQS) Averages
- NAAQS average values (8-hour averages for ozone and CO, 24-hour averages for PM2.5)
- Daily Summary Values (each monitor has the following calculated each day)
- Observation count
- Observation percent (of expected observations)
- Arithmetic means of observations
- Max observation and time of max
- AQI (air quality index) where applicable
- Number of observations > Standard Units, where applicable
- Annual Summary Values (each monitor has the following calculated each year)
- Observation count and percent
- Valid days
- Required observation count
- Null observation count
- Exceptional values count
- Arithmetic Mean and Standard Deviation
- Site and Monitor Information
- FIPS State Code (the first 5 items on this list make up the AQS Monitor Identifier)
- Site Number (unique within the county)

- Latitude
- Longitude

3.3 Data Pre-processing

Prior to training a neural network model, data preparation and pre-processing involves rescaling input and output variables using techniques like normalization and standardization. Normalization is the process of rescaling data from its original range so that all values fall between 0 and 1. The following procedures were used to pre-process the characteristics of the input data and output variable using Standard Scaler.

Use the available training data to fit the scaler. The training data will be utilized to determine the lowest and maximum observable values for normalization. The fit() method is used to do this. Apply the scale to the data collected throughout the training. As a result, the model can be trained using the normalized data by calling the transform() method. Moving forward, use the scale with data.

After training the model and predicting on the test data, we inverse scaled the features back to the original values.

3.4 Methods

3.4.1 Introduction to Machine Learning

Machine learning leads to variations in network systems that perform tasks that are connected to and linked to artificial intelligence systems. Recognition, diagnosis, forecast planning, and a robot control system are among the tasks which are performed using Machine

Learning. We may assume that machine learning is the method of training a program with various algorithms in order to bring it to the test in terms of automated intelligent data processing. Machine Learning, for example, can assist in detecting handwriting or identifying faces in a picture. Artificial intelligence gave birth to the world of machine learning. Its aim is to create smarter and more intelligent devices. The only way to do this is for the computer to learn from itself. Machine learning has emerged as a distinct skill. Machine learning now operates in so many areas of technology that we aren't even aware of it when using it. Machine learning is the analysis of algorithms in order to improve the efficiency of machines by measuring and training them with various data. In the complex model, machine learning improves and evolves principles that help it process similar tasks and objectives more effectively. Understanding input variables and how they move into vectors is crucial while training and validating a machine learning system.

3.4.2 Artificial Neural Networks (ANNs)

Artificial Neural Networks, or ANNs, are a class of machine learning algorithms (or precisely deep learning). ANN has shown that it is a very efficient algorithm with high-performance prediction, classification, and recognition capabilities. Currently, artificial neural networks are capable of learning from experience. As a result, computer networks are capable of doing very big and intelligent tasks. ANN is a parallel mechanism that accomplishes the most difficult operations or activities in various fields of finance, manufacturing, and technology. It forecasts and detects without adding to the problem's complexity. In the middle of one input and one output, ANN has hidden layers that process the information data and forward the result to the next layer, and each layer to the next layer, until it reaches the final layer, which is the output layer. ANN is the most widely used algorithm in artificial intelligence today; it is the most common machine learning algorithm.

3.4.2.1 Neurons

Our brain is made up of a network of biochemical neurons that are linked together. Our perception, reading, breathing, and motion are all realized through a network of intertwined neural networks. Some of these neuronal systems were present at birth, while others were formed by experience. Artificial Neural Networks carry out these massive and complex functions that are fed into the layers of the neural network and process the data in the same way that neurons in human brains process information.

3.4.2.2 Structure of ANN

Neural network includes a large number of units arranged in a series of layers which are the artificial intelligence neurons.

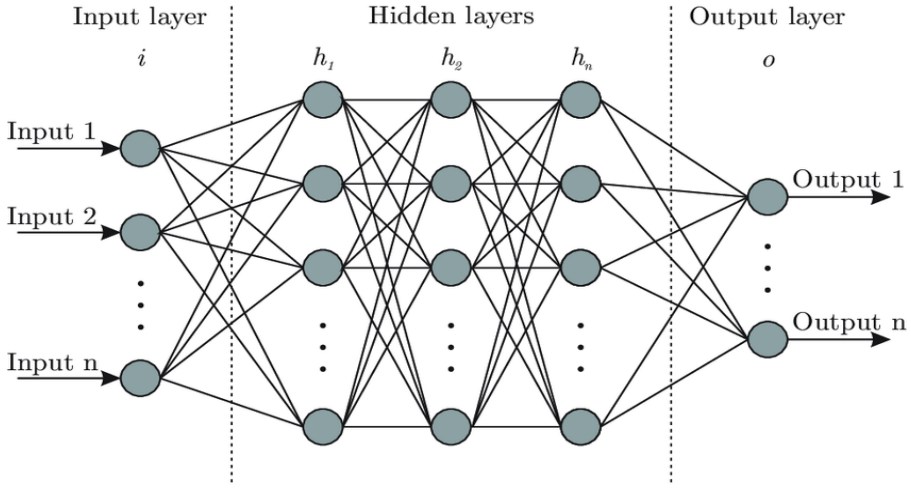


Figure 3.1 Structure of ANN [46]

As shown in Figure 3.1, Input Layer is the first layer, which includes artificial neurons that collect data from the input and learn to understand or process it. Hidden layers are the layers in the center of the input and output layers. These layers process data and convert it from the input to the output through network neurons. The weights are constantly updated to the contribution of the hidden layer for fineness and validity. The final layers apply activation on the outputs from the

hidden layer and generates the output. Most neural networks are fully connected, that means the hidden layer is fully linked between each neuron in the next output layer and to the previous layer or input layer at first.

3.4.2.3 Back-Propagation Neural Networks (BPNN)

In this section, we will be explaining how the Back-Propagation Neural Networks works and explain the training mechanism of the Back-Propagation Neural Networks. To comprehend the BPNN training phase, we must first understand the fundamental operations of neural networks. A Back-Propagation neural network's training process can be summarized as identifying weights in layers and then finding the best weights in completely connected layers to eliminate failure or minimize the cost functions so that the model can make correct predictions on the test dataset [41].

On the training dataset, weights are randomly selected in the fully connected layers in a forward process called forward propagation and mean squared error is calculated using the loss function. Following that, in a backward mechanism known as backpropagation, these weights are modified using the loss function by the gradient descent optimization algorithm. Gradient descent and backpropagation are used to update the weights and bias, with the chain rule of calculus being used to quantify the weight changes that will decrease the loss function. So, while performing backpropagation in the training of the neural networks, the cost function finds the mean squared error and the gradient descent optimization method works together.

Backpropagation is used to train the network in three steps.:

- The feedforward of the input training pattern.
- Calculate associated error with backpropagation.
- Weight adjustment.

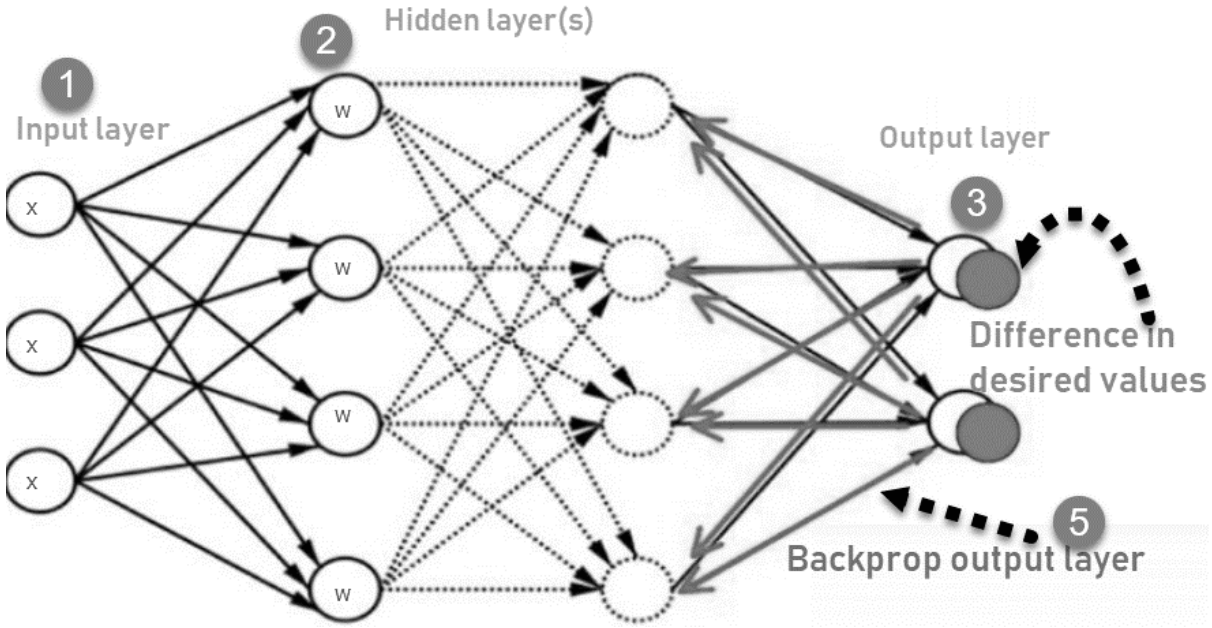


Figure 3.2 Anatomy of a Back-Propagation Neural Network [47]

Before using the backpropagation algorithm, weights are selected at random from the input layer to the hidden layers, with values ranging from –0.5 to 0.5. In addition, the data which is passed through the input layer is first normalized to avoid unnecessary over-calculations and overfitting during training [41]. The error is measured at the output layer using the mean squared error as a loss function in forward propagation on the training collection of data using the weights of the layers and the parameters of the model. Next, these weights are updated using the loss by the gradient descent optimization algorithm in backward mechanism which is called backpropagation. To measure the weight changes that will minimize the loss function, gradient descent and backpropagation are used to calculate the adjustments to weights. So, in this work backpropagation was used to train the neural networks, where mean squared, error was calculated using the cost function and gradient descent was used for optimization.

The weights of the nodes and the biases are adjusted in the backpropagation algorithm, and weights are modified based on measures of how each data point influences the weight kernel and hence the loss function. Backpropagation updates the weights and biases of hidden layers of neural networks. When we provide our back-propagation neural network input functions, the loss or error is determined in the last layer by comparing scores to truth values. The weights are updated by back propagating the measured loss to the next layer, and the process begins.

It can be assumed that the inputs are transformed into function maps in the forward process, and intermediate values such as kernel weights, strides, and so on are passed entirely through the back-propagation neural network. Loss or errors are calculated when the gradients are back propagated and each layer must obtain and return a gradient in that order during the backward propagation process. It will receive the loss gradient in relation to its output and return the loss gradient in relation to its inputs.

Weights are updated using the loss by the gradient descent optimization algorithm in backpropagation. The weights and bias are adjusted using gradient descent and backpropagation to calculate the adjustments to weights that will minimize the loss function. So, while performing backpropagation in the training of the neural networks, the cost function which is mean squared error as well as the optimization method which is gradient descent works together.

The training of the network by backpropagation algorithm can be classified into the following three stages:

- The feedforward mechanism of the input to output layer with the initialization of weights and the biases given the specific parameters and architecture of the model.
- The calculation of the mean squared error using the loss function.
- Using Gradient Descent optimization algorithm to adjust the weights and retrain the

process until the loss is minimized during backpropagation.

Firstly, before the application of the backpropagation algorithm, weights are initialized which are randomly chosen from the input layer to hidden layers generally having the values between –0.5 & 0.5. Also, the data which is passed into the input layer is firstly normalized to avoid over-calculations and unnecessary overfitting in training [41].

In the backpropagation algorithm, two adjustments are performed for weights of the layers and for the biases, weights are updated based on the measurements how each pixel affects the weight kernel and hence the loss function. In convolutional neural networks, weights are convolution kernels, and values of kernels are updated in backpropagation on convolutions. When we pass input features to our convolutional neural network, in the last layer the loss or error is calculated error by comparing scores with truth values. This calculated loss is back-propagated to the next layer to update the weights and the process continues.

It can be concluded that in a forward phase, the inputs are converted into the feature maps and are passed completely through the convolutional neural network and each layer will convolute inputs, intermediate values such as kernel weights, strides etc. calculating the loss or error while during the backward propagation phase, the gradients are back-propagated and each layer has to receive a gradient and also return a gradient in that order. It will receive the loss gradient in relation to its output and return the loss gradient in relation to its inputs.

3.4.3 Recurrent Neural Network (RNN)

Other deep learning approaches are used to solve prediction issues such as gradient descent algorithm is generally very slow because it requires small learning rates for stable learning and picking the learning rate for a nonlinear network is a challenge in addition to BPNN and ANN

models (e.g., RNN models, LSTM models and GRU models). RNN, LSTM, and GRU models have different inputs than traditional Neutral Network models because each neuron in the NN's input layers is a single sequence element, but each neuron in the RNN, LSTM, and GRU input layers is a vector encoded by the previous sequence elements. The concentrations of PM2.5 and PM10 show a significant temporal connection. In comparison to NN models, RNN, LSTM, and GRU are used to forecast PM2.5 or PM10 concentrations since they are specialists in dealing with time series data. In learning long-time dependency, LSTM, is an extended model of RNN, differs from RNN due to the phenomena of gradient vanishing in RNN. The internal structure of GRU, an expanded model of LSTM which has three gates whereas GRU has just two. In the training stage of the RNN, LSTM, and GRU models, a specified time steps-based data is taken as input, as opposed to the ANN model, which uses just meteorological data.

Core functionality of RNNs is to process sequence of input data $x_T = x_1, \ldots, x_t$ with the time step T ranging from 1 to t (see Figure 3). h_t is a hidden state at time t and also work as memory of the network.

$$h(t) = f\,(w_x\, x_t + w_h\, h_{t-1})$$

where the function f is nonlinear transformation function, e.g., tanh, ReLU. w_h is weight matrix between two hidden connections h_{t-1} and h_t, w_x is weight matrix between input x_t and hidden connection h_t.

y_t is output at time t and it is defined as an output sequence by the following equation:

$$y_t = c + w_y\, h_t$$

where c is a constant and w_y is a weight matrix between hidden connection h_t and output connection y_t.

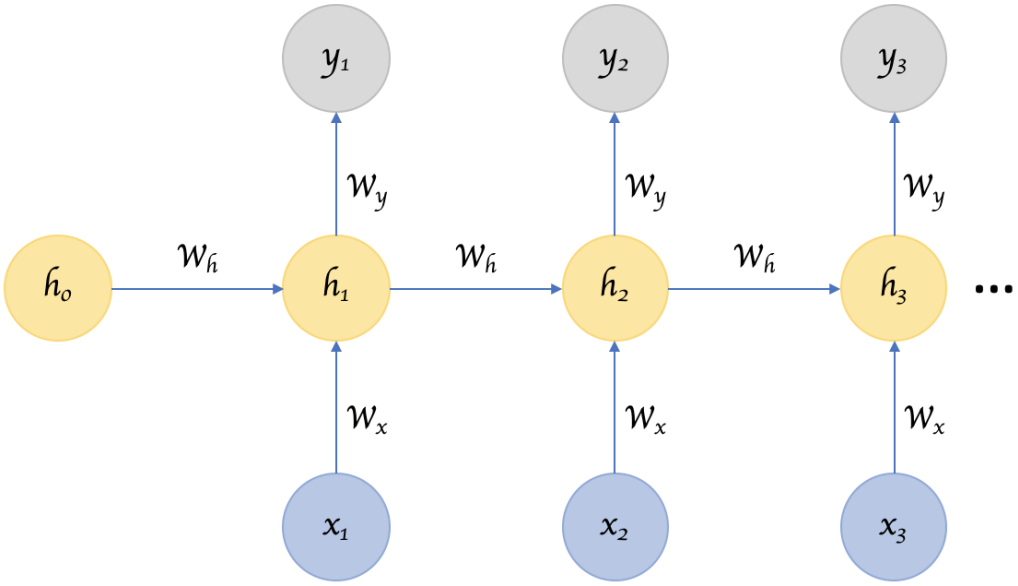


Figure 3.3 Anatomy of a Recurrent Neural Network [48]

3.4.4 Long Short-Term Memory (LSTM)

Long Short-Term Memory, or LSTM, is a form of Recurrent Neural Network that has been recognized for its ability to process time series data in the short term, according to Shaikh et al (2019). Furthermore, as compared to other deep neural networks, LSTM has shown superior learning ability in terms of temporal series [42]. Long Short-Term Memory can be thought of as a system that uses a cell state and a carry state to keep information in the shape of a gradient that is guaranteed not to be lost as the input sequence is stored deeper in the architecture. At each time step, the LSTM considers the current state, the carry state, and the cell state. Figure 3.4 shows the architectural anatomy of the LSTM network.

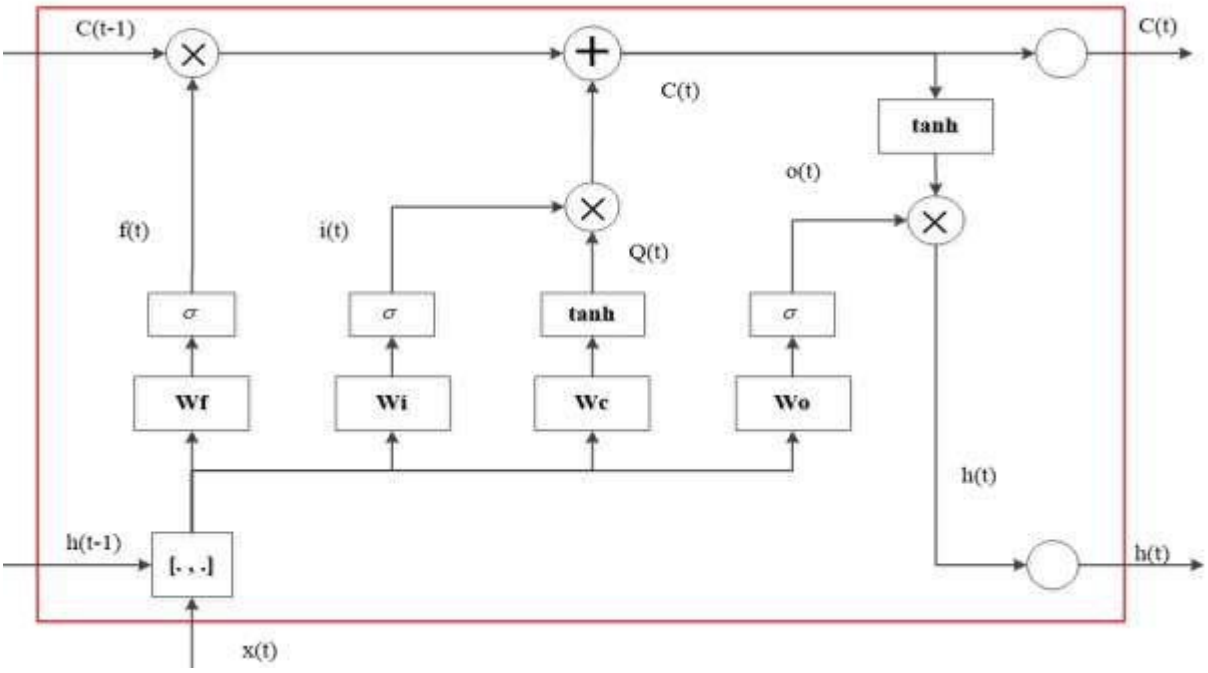


Figure 3.4 Anatomy of a Long Short-Term Memory Neural Network [49]

Long Short-Term Memory has three common gates and weight vectors. The forget gate f(t) is in control of forgetting non-essential information, the input gate i(t) is in control of working with the current input, and the output gate o(t) is in control of generating the predictions for each time point. It's worth noting that, as we showed in the Back-Propagation Neutral Network training phase, the role of each cell element is determined by the weights of the network layers, which are initialized and adjusted during training. Before moving to the discussion of the results, we need to review more general attributes of neural networks which are Cost function, a loss function (see Section 3.4.7) which has to be minimized to make the model more accurate with its predictions, gradient descent optimization algorithm (see Section 3.4.8) which updates the weights during the back-propagation, and regularization methods (see Section 3.4.9).

3.4.5 Bi-Directional LSTM

Bidirectional LSTMs are a kind of LSTM that can increase model performance when dealing with sequence issues. Bidirectional LSTMs train two instead of one LSTM on the input sequence in cases where all time steps of the input sequence are available. The hidden state of an LSTM maintains information from inputs that have already gone through it.

Because the only inputs it has seen are from the past, the information preserved by the unidirectional LSTM is limited to the past. When using bidirectional processing, the inputs will be processed in two directions, one from the past to the future and the other from the future to the past. The LSTM that runs backwards saves information from the future, while the LSTM that goes forwards saves information from both the past and the future by combining the two hidden states at every moment in time.

Only the historical context may be used by a typical LSTM network. The lack of future context, on the other hand, may result in an inadequate comprehension of the text's meaning. A forward LSTM layer and a backward LSTM layer are combined in BiLSTM. By combining the information from two gates before and after the word, the context information may be completely used. The model structure is shown in Figure 3.5.

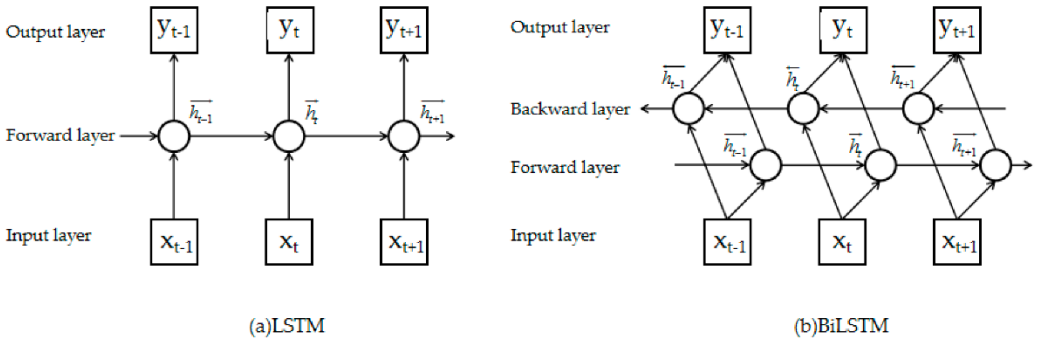


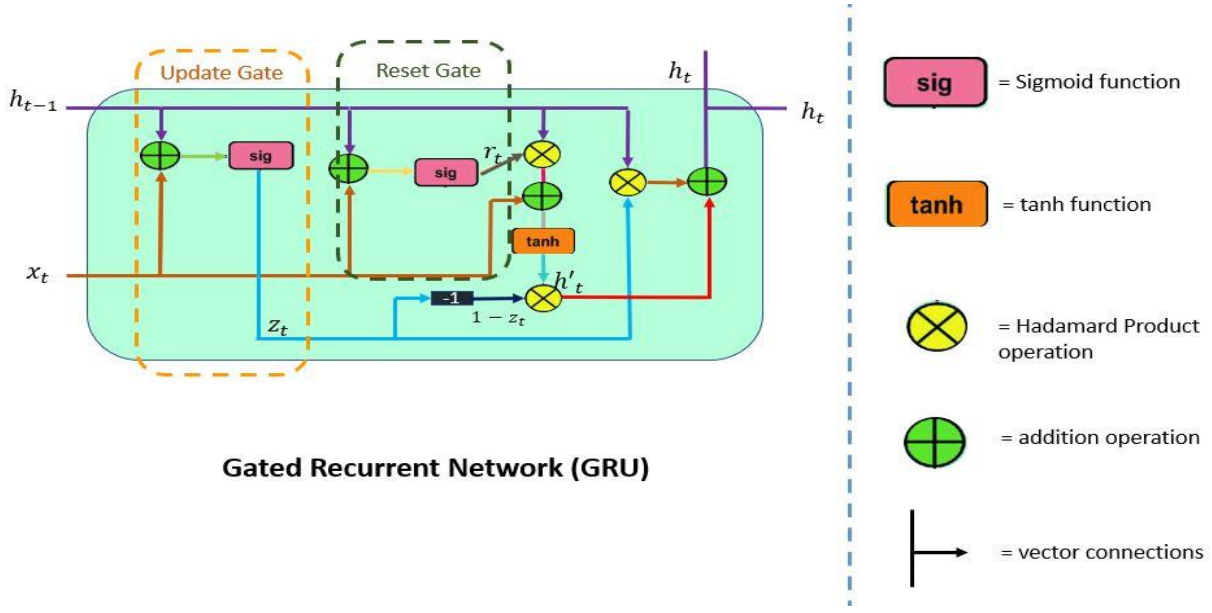
Figure 3.5 Anatomy of a Bi-Directional Long Short-Term Memory [50]

3.4.6 Gated Recurrent Unit (GRU)

As shown in Figure 3.6, a Gated Recurrent Unit (GRU) is an LSTM variation with a simplified design. GRU abandoned the cell state in favor of using hidden state to transfer information. On top of that GRU has two gates: Update Gate and Reset Gate. The update gate functions similarly to the LSTM's forget and input gates. It chooses what data to discard and what data to include, whereas the reset gate indicates whether or not the previous hidden state should be disregarded. The output of update gate is z_t and the output of the reset gate is r_t. The detailed operations of the GRU unit are illustrated in the equations below.

$$
\begin{aligned}
Z_t &= \sigma(W_z \cdot [s_{t-1}, x_t] + b_z) \\
r_t &= \sigma(W_r \cdot [s_{t-1}, x_t] + b_r) \\
\tilde{s}_t &= \tanh(W_s \cdot [r_{t-1}, x_t] + b_s) \\
s_t &= (1 - z_t) \cdot s_{t-1} + z_t + \tilde{es}_t
\end{aligned}
$$

where W_z, W_r and W_s are weight matrices; and b_z, b_r and b_s are bias vectors.



Gated Recurrent Network (GRU)

Figure 3.6 Anatomy of a Gated Recurrent Unit [51]

3.4.7 Cost Function

The forward pass during the training is responsible for the computation of loss, which is achieved through a Cost Function while during a backward pass, the weights are updated using the gradient computation. So, cost function plays a very important role in the training process of the Convolutional Networks or any other ANN algorithm. The training of an ANN which involves the reduction of error is extremely slow and time costly. The Cost function helps us to solve this problem by calculating the mean squared error which is used during the backpropagation. So, in order to make the training process faster instead of minimizing squares of the differences between of the true labels and predicting outcomes, mean squared error function is minimized or optimized. A Cost function or Loss function calculates the differences between the predictions and the ground or truth labels to give a numerical value of the cost using forward propagation. For each propagation, the loss is calculated, the forward-propagation algorithm achieves it by measuring the partial derivatives with regards to the parameters of the Neural Network. The cost function for the Convolutional Neural Networks is not specifically different and also the way CNNs are used is not unique as it works on the same processes and principles that other Neural Network architectures work. It has to be understood that the cost function is defined by the task such as regression or classification, not specifically by the architecture of the model. For example, in our case, for a regression the mean squared error will be used to calculate the distance between the output of the activation function and the ground truth label [43]. The cost function which is mean squared error is defined by the following equation-

$$\frac{1}{n}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Where, y_i is true value and $\hat{y}_i$is predicated value.

3.4.8 Gradient Descent Optimization

As discussed above, Cost function is used to calculate the errors in the forward pass and to minimize the computed errors, Gradient Descent Optimization algorithm is used to update the weights and retrain the model during back propagation using updated weights.

Over the years, researchers have been working on the improvement of new optimization algorithms as Deep Neural Networks are significantly an optimization problem where the main objective is to find global minimum using a computationally time efficient convergence process which can be accomplished using gradient descent optimization algorithm [43]. The main objective of the neural network training and optimization using Gradient Descent is to find the best values of the parameters providing a best-fitted model with least error and correct predictions for input features to the target features. A general optimization process can be mathematically denoted as the equation shown below [43] [44]:

$$f: \mathbb{R}^n \rightarrow \mathbb{R}, \text{Find } \hat{\boldsymbol{x}} = \operatorname{argmin} f(\boldsymbol{x}), \boldsymbol{x} \in \mathbb{R},$$

Where, f is the cost function.

Gradient descent is a widely used optimization algorithm in neural networks which can continuously adjust or update the learning parameters of the training model which includes kernels and weights and hence minimizing the computed loss or error term. Figure 3.7 shows the graph of the loss function as a function of the learnable parameter w.

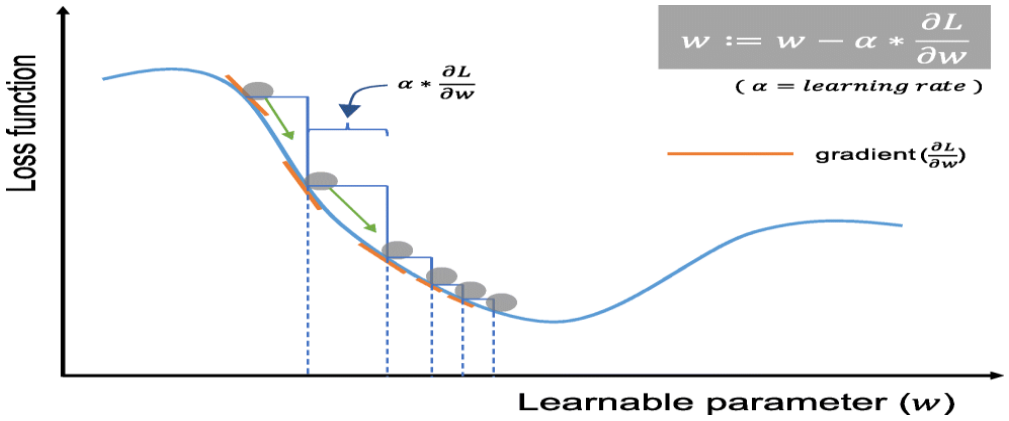


Figure 3.7 Gradient Descent Optimization Algorithm [53]

Specifically, the gradient descent is given by the partial derivative of the cost with respect to each parameter which has to be learned during the training, and a single update of a learning parameter can be mathematically written as:

$$\mathbf{w: = w - \alpha * \partial L/\partial w}$$

Where w stands for a parameter to be optimized, α is a learning rate, and L is a loss function.

Most of the time, because of memory constraints, the gradients of the cost function are computed by a small subset of the training dataset rather than full dataset which is called a Mini-batch gradient descent method, also known as stochastic gradient descent or SGD where mini-batch size is also a hyper-parameter which can also be optimized for better results. Many improvements have been proposed on stochastic gradient descent optimization algorithm which are widely used. Some of them are stochastic gradient descent with momentum, Root Mean Square Propagation (RMSprop) and Adaptive Moment Learning (Adam).

Due to the speed and effectiveness in dealing with large-scale optimization problems, first-order based optimization algorithms play a major role in deep learning. Adam optimization

learning algorithm is a first-order gradient-based optimization algorithm having low memory requirements and is computationally efficient for problems which require large number of parameters.

Stochastic gradient descent (SGD) is the typical technique for training deep neural networks in these methods, in which parameters are updated iteratively in the direction of the negative gradient until convergence. Following that, various SGD versions that may adjust the learning rate by itself by employing the square root operation of some type of averaging of the squared components in the past gradients were developed [44][45]. Adagrad was the first variant which worked well with sparse data. This technique, on the other hand, utilizes all previous gradients, which might cause the learning rate to rapidly decrease. Some algorithms, such as Adam, Adadelta and RMSprop were developed to overcome this problem by employing the exponential moving average of past squared gradients. Adam optimization algorithm is a heuristic of both Momentum-based and Root Mean Square Propagation (RMSProp). Adam is an adaptive learning rate method which is a combination of Stochastic Gradient Descent (SGD) with momentum and RMSprop. It calculates the squared gradients for scaling and alters the learning rate like RMSprop and also utilizes the moving average of the momentum of gradient in a similar way as in SGD.

3.4.9 Regularization

A neural network or any other complex network such as CNN, LSTM etc. consists of a large number of learning parameters and hence chances of overfitting is a serious concern in such networks. These networks are also time costly in training which makes overfitting a serious problem as to achieve a better performing model in the desired timeframe. So, the overfitting in

the training of a neural network can occur very easily and is dependent on various factors. Regularization is a method with which we can make model to train itself in a more generalized way which also improves model's performance on the unseen data. There are two regularization methods used in this research, namely L1 and L2 regularization in the NN layers and dropout layers in the architecture of the model. As in the machine learning algorithms, the regularization penalizes the coefficients to avoid overfitting, in a similar manner, in deep learning, the regularization penalizes the weights of layers at each node.

The most popular regularization methods in neural networks are L1 and L2 which adds a regularization term to the cost function to update as the following equation:

Cost function = Loss (mean squared error) + Regularization term

Since a neural network with lower weight matrices leads to simpler models, the addition of regularization component results in simpler models as it reduces the weight of the matrices. As a result, it will significantly decrease overfitting.

Dropout is a very essential regularization technique and is widely used in complex networks such as LSTM. The dropout approach, which gives excellent results, is the most often used regularization technique. A dropout layer selects certain nodes at random and eliminates them together with all of their incoming and outgoing connections at each iteration, reducing the complexity of a highly complex model. At each iteration, different set of nodes are selected resulting in dropping some nodes in a different set of outputs. Dropout performs better in a LSTM, or any other complex architecture by zeroing out the activation values of randomly picked neurons during the training phase, resulting in no overfitting and a more robust model. Instead of depending on the predictive abilities of a small selection of neurons in the network, this constraint on the training process allows the extremely complex network to acquire more robust features.

3.4.10 Fully Connected Layer and Time Distributed Layer

The fully connected layer, also known as a dense or feed-forward layer, is the most general-purpose deep learning layer. This layer has the least amount of structure of all the layers and is found in almost all neural networks. It's commonly used to control the output layer's size and shape.

On the other hand, RNNs, including LSTMs employ a Time distributed dense layer to maintain one-to-one input and output relationships. As training convolutional flow for each sample of the sequence is time-consuming and ineffective for learning unique characteristics, a standard sequential neural network is not suited for this job. To solve this challenge, a module that can apply the same layer to a list of input data is required. Keras includes a Time Distributed layer object that aids in the detection of intents hidden behind chronological inputs. Every temporal slice of the input is given a layer. It aids in maintaining one-to-one relationships between input and output. The flattened output of the model is intermingled with the time steps if a Time Distributed Layer is not utilized for sequential data. However, if this layer is used, the output for each time step is produced individually. Recurrent Neural Networks with LSTM are a more powerful form of RNNs. RNNs may handle a variety of input/output configurations, including one-to-one, one-to-many, many-to-many, and many-to-one. Because the outputs of one-to-many and many-to-many architectures should have the same function for every time step, the Time Distributed Layer (and its predecessor, the Time Distributed Dense Layer) is frequently utilized. If this isn’t the case, the network’s output will be flattened. The flattened output would prevent time step data from being separated, resulting in unnecessary intervention between time steps.

3.4.11 Activation Function

Activation function is provided in almost all the layers, which transforms the data or features. The most important activation functions are:

- Linear activation function – no change in output.
- Rectified Linear Unit – output is 0 when input is less than 0 else same as input.
- Sigmoid – output range (0,1)
- Tanh – output range (−1,1)

Table 3.3 shows all the activation functions.

Table 3.3 Activation Functions

Activation Function	Equation	Range
Linear Function	$f(x) = x$	$(-\infty, \infty)$
Step Function	$f(x) = \begin{cases} 0 \; for \; x < 0 \\ 1 \; for \; x \geq 0 \end{cases}$	{0,1}
Sigmoid Function	$f(x) = \sigma(x) = \frac{1}{1 + e^{x}}$	(0,1)
Hyperbolic Tangent Function (tanh)	$f(x) = tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$	(−1,1)
Rectified Linear Unit (ReLU)	$f(x) = \begin{cases} 0 \; for \; x < 0 \\ x \; for \; x \geq 0 \end{cases}$	$[0, \infty)$

The activation function's output is sent as a prediction or probability to all neurons (also known as nodes or units) in the following layer or in the output layer.

3.4.12 Keras Tuner – hyper-parameter Optimization

A Hyper parameter Optimization process is simply defined as the process which is used to get the best model based on the best combinations of the parameters in an algorithm. There are a number of parameters which defines an algorithm and can accept different values. The different values of the parameter affect the performance of the model, whether they are machine learning models or deep learning models. In deep learning, the number of training iterations, batch size, activation function, learning rate, and optimizer algorithm are some of the parameters that can be tuned. There are other core hyper-parameters that can be tuned for particular architectures like LSTM, such as the number of filters in the layer, the size of the filter, and the stride. The number of units in the layer and whether or not to return sequence are typical hyper-parameters in LSTM that should be tuned. However, they are not just numerical numbers, but also categorical values, such as whether the optimizer is a Momentum SGD or Adam. Without hyper-parameter tuning, it's nearly impossible for a machine-learning algorithm to accomplish the job. Deep learning tends to have a high number of hyper parameters and tuning of these hyper parameters has a big impact on the performance. Table 3.4 gives the deep learning frameworks for hyper-parameter optimization. For parameter optimization, we suggest using Keras-Tuner, a next-generation hyper-parameter tuning framework. KerasTuner is a recently created open-source hyper-parameter tuning optimization with a large capacity to automate the fit-trial-and-error-calculation process of hyper-parameters optimization. Keras-Tuner uses a focused method to discover the best or optimal hyper-parameter values, which can be expressed in an API style and is also quite compact when compared to other frameworks.

Table 3.4 Deep Learning Frameworks for Hyper-parameter Optimization

	Deep Learning Frameworks	**Hyperparameter Optimization Frameworks**
Define-and-Run Style (Symbolic, static)	Torch (2002), Theano (2007), Caffe (2013), TensorFlow (2015), MXNet (2015), Keras (2015)	SMAC (2011), Spearmint (2012), Hyperopt (2015), GPyOpt (2016), Vizier (2017), Katib (2018), Tune (2018), Autotune (2018)
Define-by-Run Style (Imperative, dynamic)	Chainer (2015), DyNet (2016), PyTorch (2016), Keras (2015) TensorFlow Eager (2017), Gluon (2017), MXNet (2015), Xgboost(2014) , LighGBM(2018)	***KerasTuner (2019)***

Another advantage of Keras-Tuner is that, it can be integrated with almost all machine learning and deep learning-based frameworks in python which includes Chainer, Scikit-learn, Pytorch, MXNET, LightGBM and others. Table 3.5 compares different deep learning frameworks with the hyper-parameter tuning methods.

Table 3.5 Comparison of hyper-parameter tuning frameworks

Framework	Pruning	Lightweight	Distributed	Dashboard
SMAC	✗	✓	✗	✗
GPyOpt	✗	✓	✗	✗
Spearmint	✗	✓	✓	✗
Hyperopt	✗	✓	✓	✗
Autotune	✓	✗	✓	✓
Vizier	✓	✗	✓	✓
Katib	✓	✗	✓	✓
Tune	✓	✗	✓	✓
KerasTuner	✓	✓	✓	✓

From the above comparison, it is clear that Keras-Tuner outperforms all other frameworks which motivates us to use Keras-Tuner in our research.

CHAPTER 4

Results and Analysis

RNN, LSTM, and GRU all have parameterized functions that directly affect the data's ideal parameters, resulting in better prediction. On the designs, various studies and configurations are used to acquire the best estimates of parameters such as number of units in a layer and learning rate. We employed three different base architectures in this research, including RNN, LSTM and GRU to generate eight different models. The input layer comes first, followed by the hidden layer, and finally the output layer in all designs. The train and test data are normalized in the experiment. As part of the experimentation, a variety of parameters are applied. We used two datasets in all the experiments which are AirNet and EPA datasets as described in the further sections of this chapter. Figure 4.1 shows an overview of the architecture of the model to predict PM10 values using a combination of deep learning algorithms like RNN, LSTM, GRU.

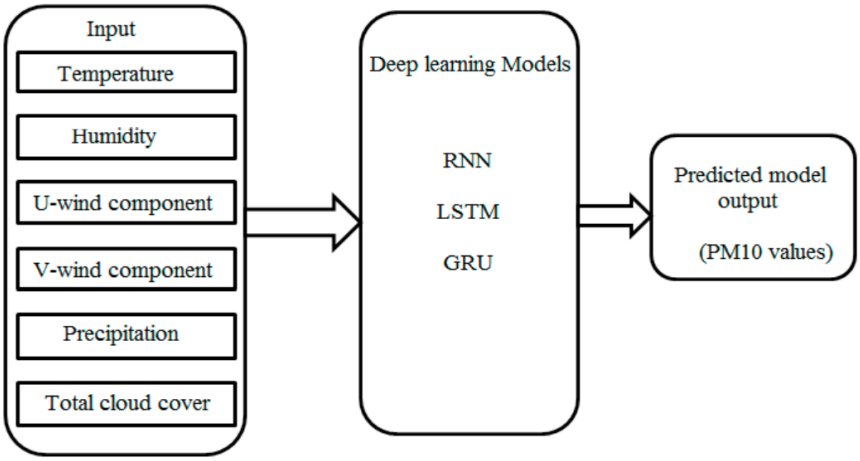


Figure 4.1 Overview of features and Labels - Air Quality Prediction using Deep Learning [54]

4.1 Deep Learning Architecture

Figure 4.2 shows the deep learning architecture used in the experiments for predicting PM10 values. Eight deep learning models were created with combinations of Fully Connected Layers (FCL), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Recurrent Neural Network (RNN), Time Distributed Layer (TDL) as middle layer with different number of neurons. Every layer has an output shape of the form (batch size, time steps, # of units). The first parameter represents the batch size of data. The second parameter represents the time steps, which is the number of values in a sequence. The third parameter represents the number of units in one input sequence. and the output of each layer also includes the total number of trainable parameters associated. In every model, a dropout layer is used to reduce data overfitting and performance improvement. Non-trainable model parameters are those that must be defined apriori or given as inputs since they will not be updated and optimized during training.

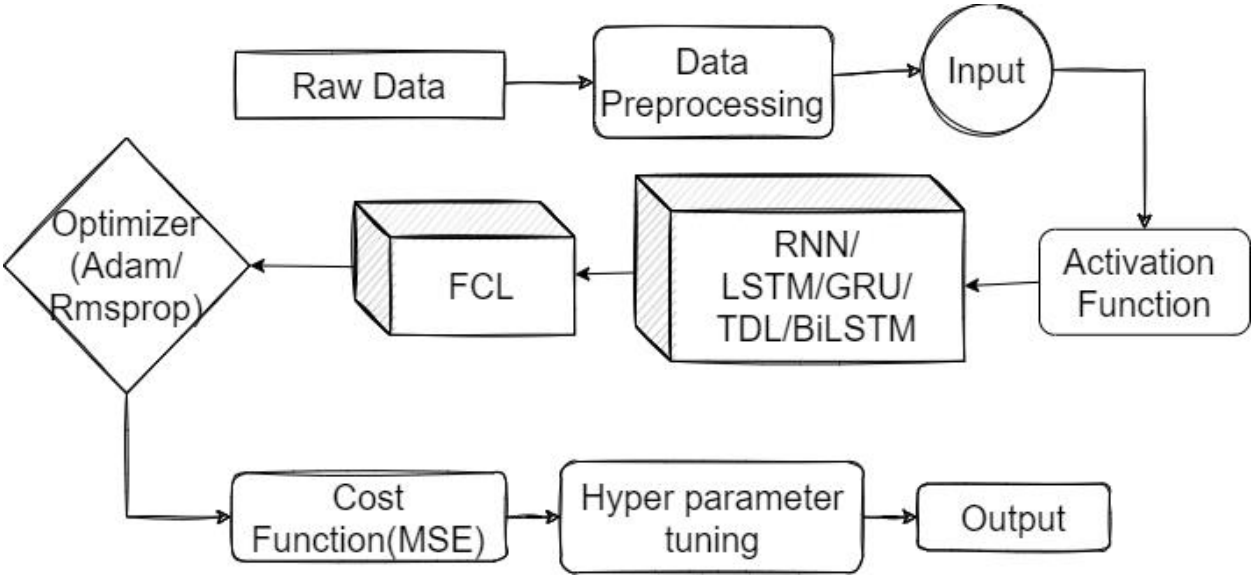


Figure 4.2 Deep Learning Architecture

4.2 AirNet Dataset Results

AirNet is made up of 6 air quality indicators obtained from 1498 sites by the China National Environmental Monitoring Centre (CNEMC), as well as meteorological data from the Global Forecasting System (GFS) [55] [56]. At each given time location in the system, each meteorological characteristic comprises 1,038,240 values. Every six hours National Oceanic and Atmospheric Administration (NOAA) releases a three-dimensional matrix to GFS. To acquire AirNet data, this three-dimensional matrix is interpolated with two-dimensional data to generate a four-dimensional matrix. The information in the four-dimensional database comes from a variety of sources (latitude, longitude, time steps and features). The information was gathered between April 1, 2015, and September 1, 2017. For each time frame, there is 6 GFS features and 7 air quality indices. The total dimension of data is (132,228,707,213). Table 4.1 shows a few observations of AirNet data set.

Table 4.1 AirNet dataset

	Temperature (K)	Relative Humidity (%)	U-wind Component (m/s)	V-wind Component (m/s)	Precipitation Rate ($\frac{kg}{\frac{m^3}{s}}$)	Total Cloud Cover (%)	PM10 ($\mu g/m^3$)
0	275.90	82.70	-3.08	1.56	0.00	0.00	31.00
1	281.60	64.80	-3.90	2.76	0.00	0.00	31.00
2	286.60	40.30	-4.47	6.51	0.00	2.00	31.00
3	288.50	35.10	-5.04	6.75	0.00	1.00	31.00
4	286.80	36.50	-4.78	5.21	0.00	0.00	31.00

Figure 4.3 shows the distribution of few observations of the target variable which is PM10.

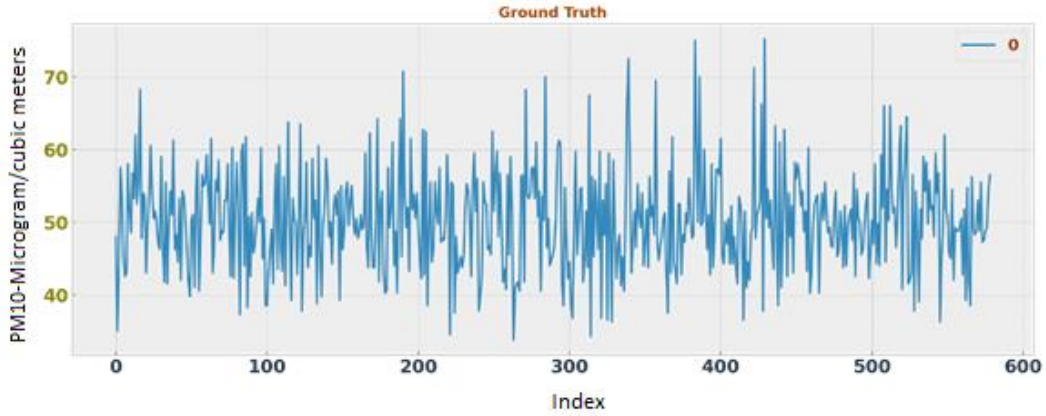


Figure 4.3 Distribution of few observations of PM10

4.2.1 Model 1 - LSTM (128)-GRU (64)-GRU (32)-FCN (64, 32, 1)

Table 4.2 shows the architecture of Model 1 with 128 units of LSTM, followed by 64 units of GRU, followed by another layer of 32 Units of GRU and this is followed by three layers of FCL with 64, 32, 1 Units. There are three dropout layers, one after LSTM and second and third after GRU layers. Table 4.2 shows the output shape received at the respective layer and the total number of trainable parameters associated with the layer.

Table 4.2 Architecture of Model 1

Layer (type)	Output Shape	Parameters
LSTM	(None, 4, 128)	69120
Dropout	(None, 4, 128)	0
GRU	(None, 4, 64)	49408
Dropout	(None, 4, 64)	0
GRU	(None, 32)	9408
Dropout	(None, 32)	0
FCL 1 (Dense)	(None, 64)	2112

```
FCL 2 (Dense)                (None, 32)                2080
_________________________________________________________________
FCL 3 (Dense)                (None, 1)                 33
=================================================================
Total Parameters: 132,161
Trainable Parameters: 132,161
Non-trainable Parameters: 0
=================================================================
```

The evaluation parameters Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set obtained are 0.3376 and 0.1008 respectively. Figure 4.4 shows the true and predicted values for some test observations. From the figure we can see that there is a difference between true values and predicted values which is noticeable.

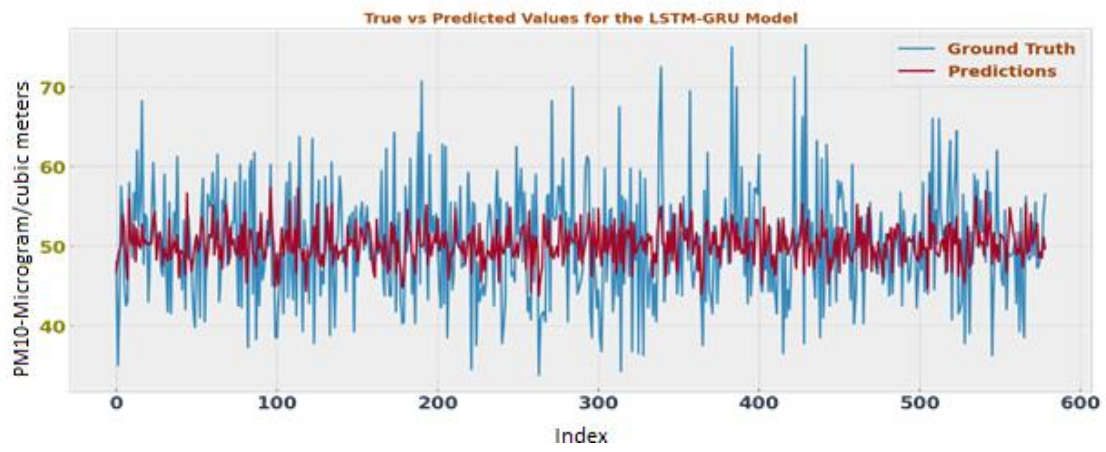


Figure 4.4 PM10 of the Ground Truth and the predicted values of Model 1

4.2.2 Model 2 - RNN (128)-RNN (32)-FCL (64, 16, 1)

Table 4.3 shows the architecture of Model 2 with 128 units of RNN, followed by another layer of 32 Units, followed by three layers of FCL with 64, 16, 1 Units. There are two dropout layers after each RNN layer. Table 8 shows the output shape received at the respective layer and the total number of trainable parameters associated with the layer.

Table 4.3 Architecture of Model 2

Layer (type)	Output Shape	Parameters
Simple RNN	(None, 4, 128)	17280
Dropout	(None, 4, 128)	0
Simple RNN	(None, 32)	5152
Dropout	(None, 32)	0
FCL 1 (Dense)	(None, 64)	2112
FCL 2 (Dense)	(None, 16)	1040
FCL 3 (Dense)	(None, 1)	17

Total Parameters: 25,601
Trainable Parameters: 25,601
Non-trainable Parameters: 0

The evaluation parameters Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.337 and 0.1023 respectively. This model gave the worst MAPE value among all 8 models. Figure 4.5 shows the true and predicted values for some test observations. From the figure we can see that there is a difference between true values and predicted values which is noticeable.

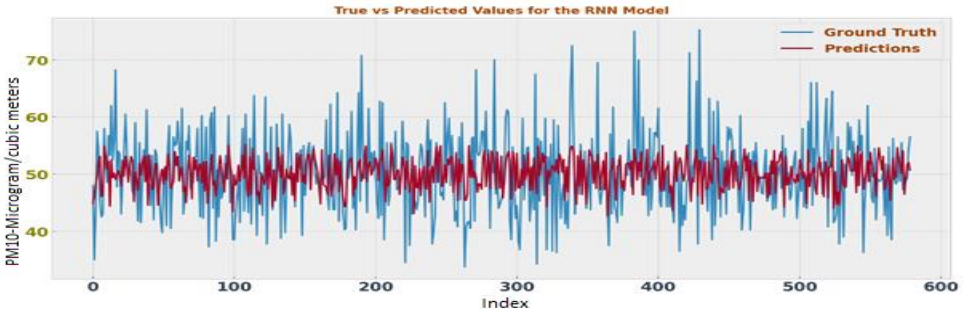


Figure 4.5 PM10 of the Ground Truth and the predicted values of Model 2

4.2.3 Model 3 - GRU (128)-GRU (64)-FCL (64, 16, 1)

Table 4.4 shows the architecture of Model 3 with 128 units of GRU followed by another layer of 64 Units of GRA, followed by three layers of FCL with 64, 16, 1 Units. There are two

dropout layers, one after each GRU layer. Table 4.4 shows the output shape received at the respective layer and the total number of trainable parameters associated with the layer.

Table 4.4 Architecture of Model 3

Layer (type)	Output Shape	Parameters
GRU	(None, 4, 128)	52224
Dropout	(None, 4, 128)	0
GRU	(None, 64)	37248
Dropout	(None, 64)	0
Dense	(None, 64)	4160
Dense	(None, 16)	1040
Dense	(None, 1)	17

Total Parameters: 94,689
Trainable Parameters: 94,689
Non-trainable Parameters: 0

The evaluation parameters Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.3012 and 0.0937 respectively. This model achieved the second best MAPE and the best RMSE values among all 8 models. So, this is the best model among all the models. Figure 4.6 shows the true and predicted values for some test observations. From the figure we can see that there is a difference between true values and predicted values which is significantly close to each other.

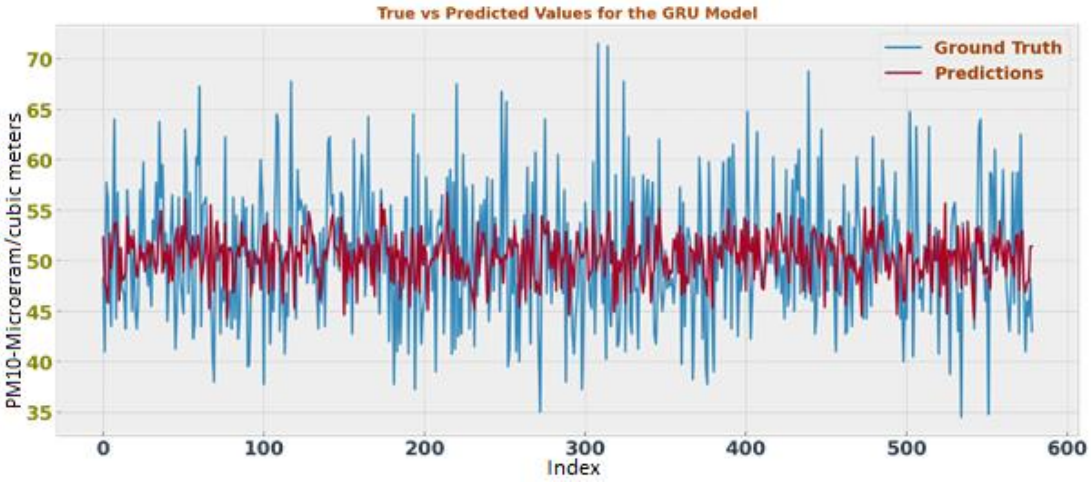


Figure 4.6 PM10 of the Ground Truth and the predicted values of Model 3

4.2.4 Model 4 - BiLSTM (32)-GRU (32)-TDL (8)-FCL (1)

Table 4.5 shows the architecture of Model 4 with 32 units of Bidirectional LSTM followed by 32 units of GRU, followed by 8 units of Time Distributed Layer (TDL) and one layer of FCL with 1 Unit. There are two dropout layers, one after Bidirectional LSTM and second after GRU layer. Table 4.5 shows the output shape received at the respective layer and the total number of trainable parameters associated with the layer.

Table 4.5 Architecture of Model 4

Layer (type)	Output Shape	Parameters
Bidirectional	multiple	9984
Dropout	multiple	0
GRU	multiple	9408
Dropout	multiple	0
Time Distributed Layer	multiple	264
FCL 1 (Dense)	multiple	9

Total Parameters: 19,665
Trainable Parameters: 19,665
Non-trainable Parameters: 0

The Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.3369 and 0.0936 respectively. This model achieved the best MAPE values among all 8 models but RMSE value is mediocre. Figure 4.7 shows the true and predicted values for some test observations. From the figure we can see that there is a difference between true values and predicted values which is significant.

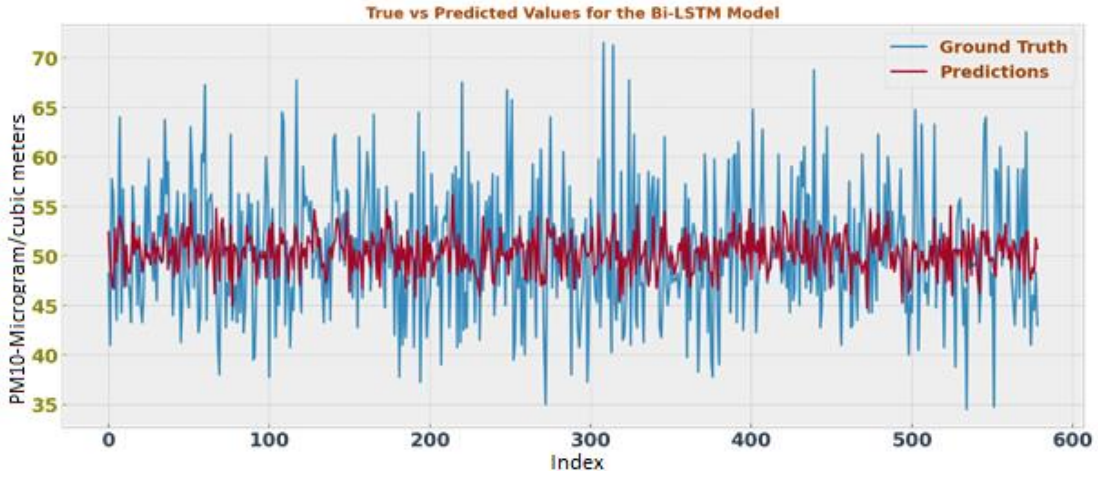


Figure 4.7 PM10 of the Ground Truth and the predicted values of Model 4

4.2.5 Model 5 - BiLSTM (64)-TDL (8)-FCL (32, 16, 1)

Table 4.6 shows the architecture of Model 5 with 64 Units of Bidirectional LSTM followed by 8 Units of Time Distributed Layer (TDL) followed by three layers of FCLs with 32,16 and 1 Units. There are three dropout layers, one after the Bidirectional LSTM layer, second after first FCL layer and third right after second FCL layer. Table 4.6 shows the output shape at the respective layers and the total number of trainable parameters associated with the layer.

Table 4.6 Architecture of Model 5

Layer (type)	Output Shape	Parameters
Bidirectional	(None, 4, 128)	36352
Dropout	(None, 4, 128)	0
Time Distributed	(None, 4, 8)	1032
FCL 1 (Dense)	(None, 4, 32)	288
Dropout	(None, 4, 32)	0
FCL 2 (Dense)	(None, 4, 16)	528
Dropout	(None, 4, 16)	0
FCL 3 (Dense)	(None, 4, 1)	17

Total Parameters: 38,217
Trainable Parameters: 38,217
Non-trainable Parameters: 0

The Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.414 and 0.0995, respectively. This model has the worst RMSE value among all 8 models. Figure 4.8 shows the true and predicted values for some test observations. From the figure we can see that there is a difference between true values and predicted values which is significant.

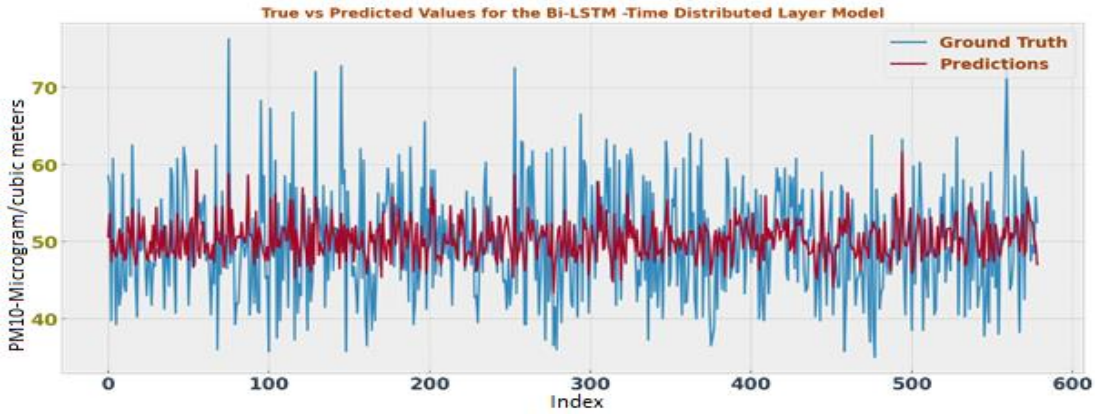


Figure 4.8 PM10 of the Ground Truth and the predicted values of Model 5

4.2.6 Model 6 - LSTM (128)-TDL (8) - LSTM (64)-FCL (32, 16, 1)

Table 4.7 shows the architecture of Model 6 with 128 Units of LSTM, 8 Units of Time Distributed Layer (TDL), 64 Units of LSTM followed by three layers of FCLs with 32,16 and 1 Units. There are four dropout layers, one after the first LSTM layer, second after the second LSTM layer, third after the first FCL layer and fourth right after the second FCL layer. Table 4.7 shows the output shape at the respective layers and the total number of trainable parameters associated with the layer.

Table 4.7 Architecture of Model 6

Layer (type)	Output Shape	Parameters
LSTM	(None, 4, 128)	69120
Dropout	(None, 4, 128)	0
Time Distributed	(None, 4, 8)	1032
LSTM	(None, 64)	18688
Dropout	(None, 64)	0
FCL 1 (Dense)	(None, 32)	2080
Dropout	(None, 32)	0
FCL 2 (Dense)	(None, 16)	528
Dropout	(None, 16)	0
FCL 1 (Dense)	(None, 1)	17

Total Parameters: 91,465
Trainable Parameters: 91,465
Non-trainable Parameters: 0

The Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.3456 and 0.0988, respectively. Figure 4.9 shows the true and predicted values for some test observations. From the figure we can see that there is a difference between true values and predicted values which is not too significant.

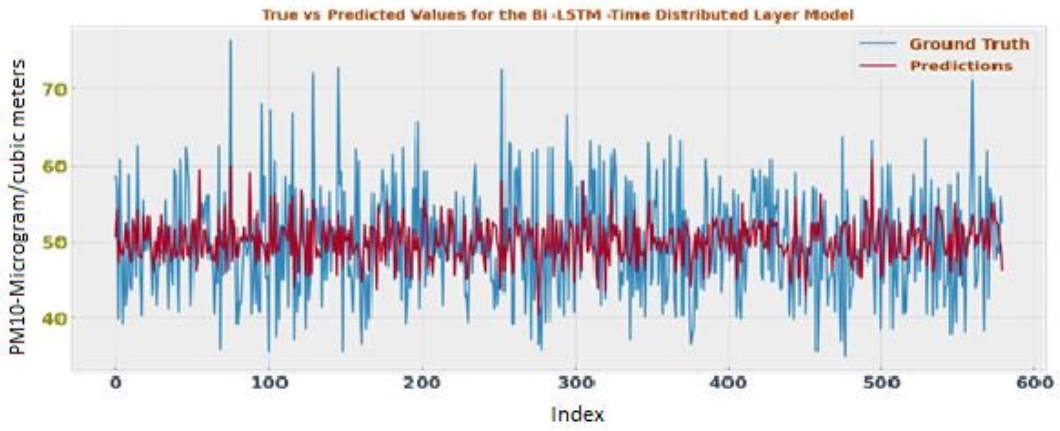


Figure 4.9 PM10 of the Ground Truth and the predicted values of Model 6

4.2.7 Model 7 - RNN (128)-TDL (Flatten)-RNN (64)-FCL (16, 1)

Table 4.8 shows the architecture of Model 7 with 128 Units of RNN, Time Distributed Layer (TDL) as a flatten layer, 64 Units of RNN and two layers of FCLs with 16 and 1 Units. Table 4.8 shows the output shape at the respective layers and the total number of trainable parameters associated with the layers.

Table 4.8 Architecture of Model 7

Layer (type)	Output Shape	Parameters
Simple RNN	(None, 4, 128)	17280
Time Distributed	(None, 4, 128)	0
Simple RNN	(None, 50)	8950
FCL 1 (Dense)	(None, 64)	3264
FCL 2 (Dense)	(None, 16)	1040
FCL 3 (Dense)	(None, 1)	17

Total Parameters: 30,551
Trainable Parameters: 30,551
Non-trainable Parameters: 0

The Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.3256 and 0.0998, respectively. Figure 4.10 shows the true and predicted values for some test observations. From the figure we can see that there is difference between true values and predicted values which is not too significant.

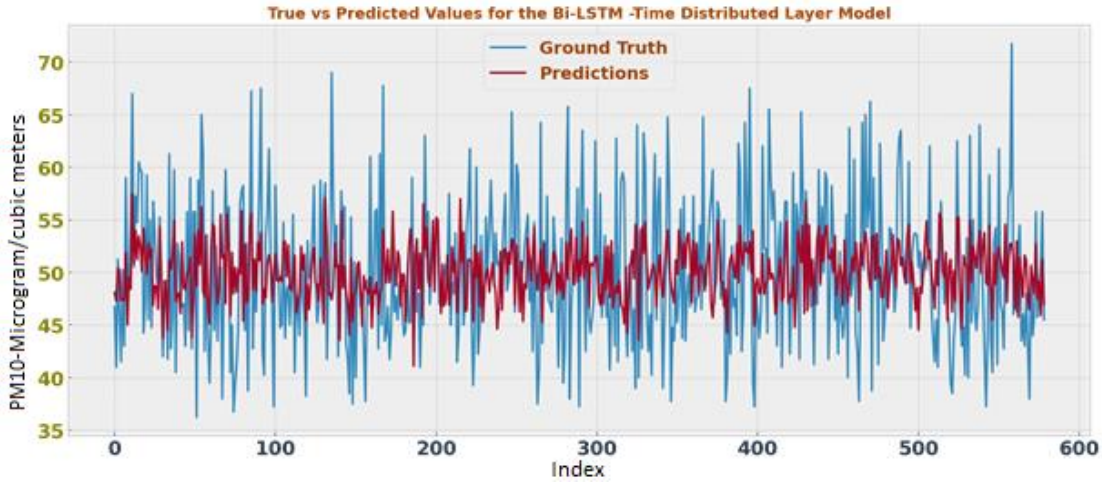


Figure 4.10 PM10 of the Ground Truth and the predicted values of Model 7

4.2.8 Model 8 - GRU (128)-TDL (Flatten)-FCL (64, 16, 1)

Table 4.9 shows the architecture of Model 8 with 128 Units of GRU, Time Distributed Layer (TDL) as a flatten layer, and three layers of FCLs with 64,16 and 1 Units. Table 4.9 shows the output shape at the respective layers and the total number of trainable parameters associated with the layers.

Table 4.9 Architecture of Model 8

Layer (type)	Output Shape	Parameters
GRU	(None, 4, 128)	52224
Time Distributed Layer	(None, 4, 128)	0
GRU	(None, 50)	27000
Dense	(None, 64)	3264
Dense	(None, 16)	1040
Dense	(None, 1)	17

Total Parameters: 83,545
Trainable Parameters: 83,545
Non-trainable Parameters: 0

The Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.3276 and 0.0999, respectively. Figure 4.11 shows the true and predicted values for some test observations. From the figure we can see that there is a difference between true values and predicted values which is not too significant and is similar to Model 7 results.

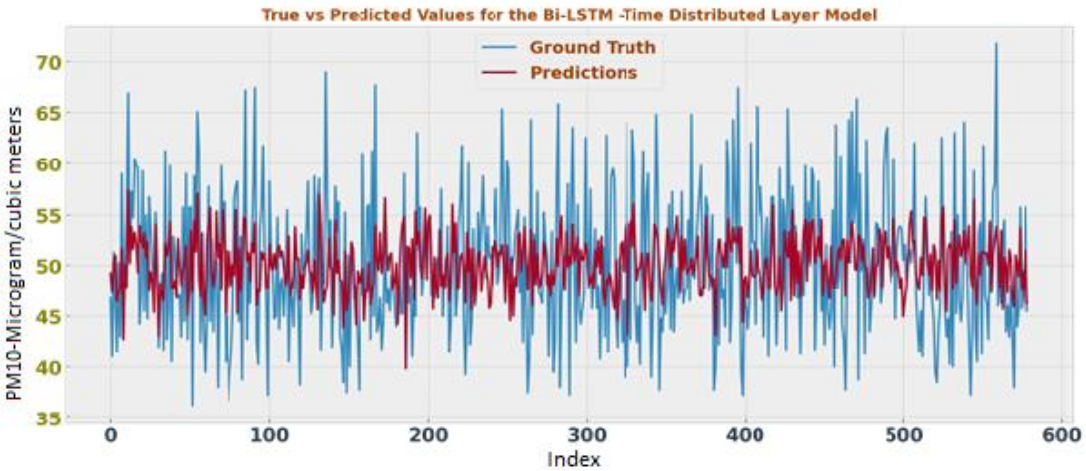


Figure 4.11 PM_{10} of the Ground Truth and the predicted values of Model 8

4.2.9 Comparison of Models on AirNet Dataset

Table 4.10 and Figures 4.12-4.14 compare all the models based on RMSE and MAPE. It can be observed that Model 3 – GRU (128) - GRU (640) - FCL (64, 16, 1) has performed the best with least value of RMSE and MAPE while the Model 5 – BiLSTM (64)-TDL (8)-FCL (32, 16, 1) has performed the worst based on the highest value of RMSE and Model 2 – RNN (128) – RNN (32) – FCL (64, 16, 1) has performed the worst based on the highest value of MAPE.

Table 4.10 Comparison of different models

Model	RMSE	MAPE
Model 1 - LSTM (128) -GRU (64) - GRU (32) - FCN (64,32,1)	0.3376	0.1008
Model 2 - RNN (128) - RNN (32) - FCL (64,16,1)	0.337	0.1023
Model 3 - GRU (128) - GRU (64) - FCL (64,16,1)	0.3012	0.0937
Model 4 - BiLSTM (32) - GRU (32) - TDL (8) - FCL (1)	0.3369	0.0936
Model 5 - BiLSTM (64) - TDL (8) - FCL (32,16,1)	0.414	0.0995
Model 6 - LSTM (128) - TDL (8) - LSTM (64) - FCL (32,16,1)	0.3456	0.0988
Model 7 - RNN (128) - TDL (Flatten) - RNN (64) - FCL (16,1)	0.3256	0.0998
Model 8 - GRU (128) - TDL (Flatten) - FCL (64,16,1)	0.3276	0.0999

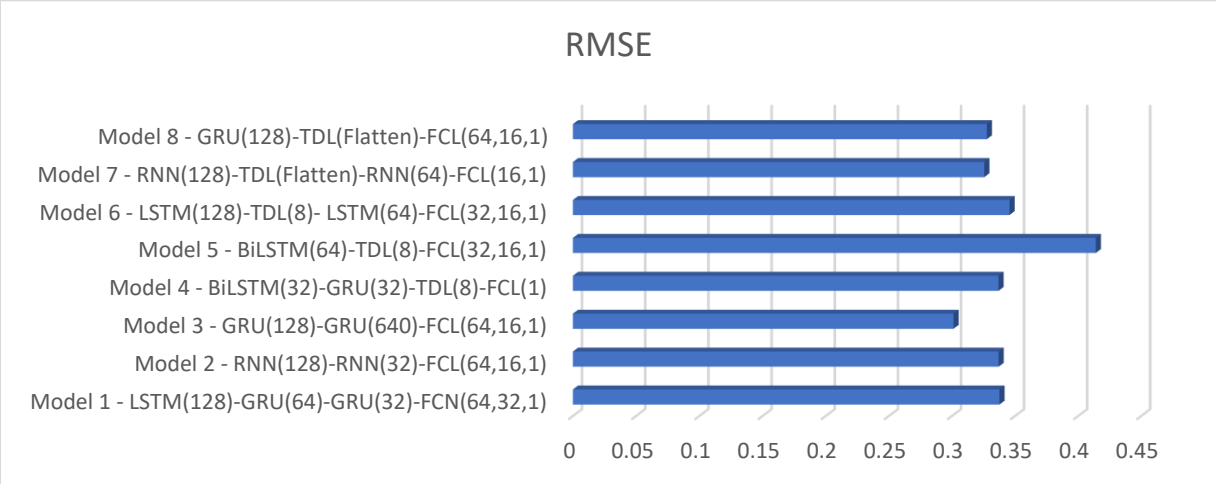


Figure 4.12 Mean Square Error (RMSE) of the different models

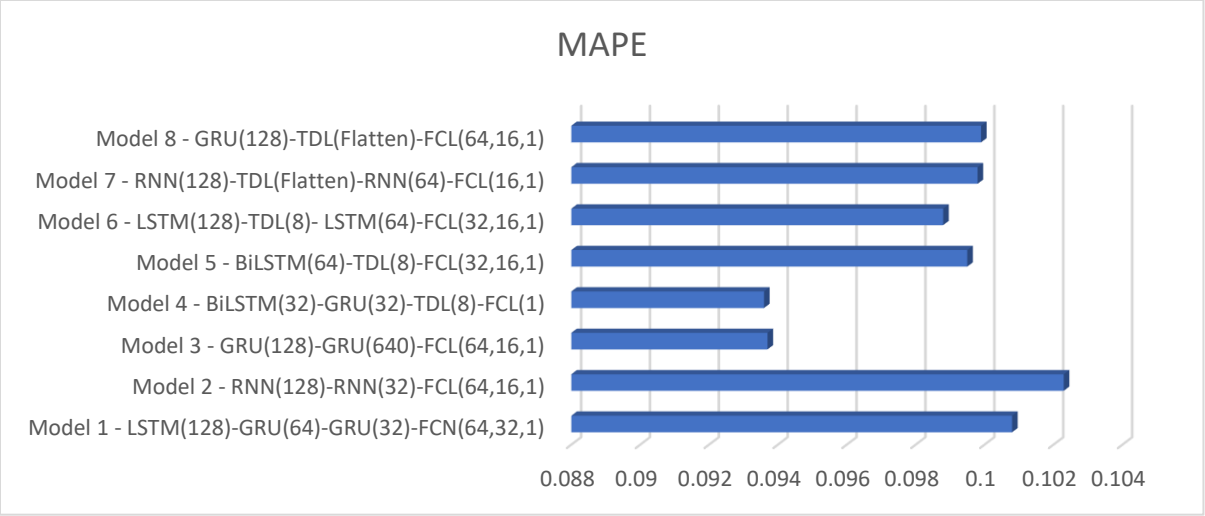


Figure 4.13 Mean Absolute Percentage Error (MAPE) of different models

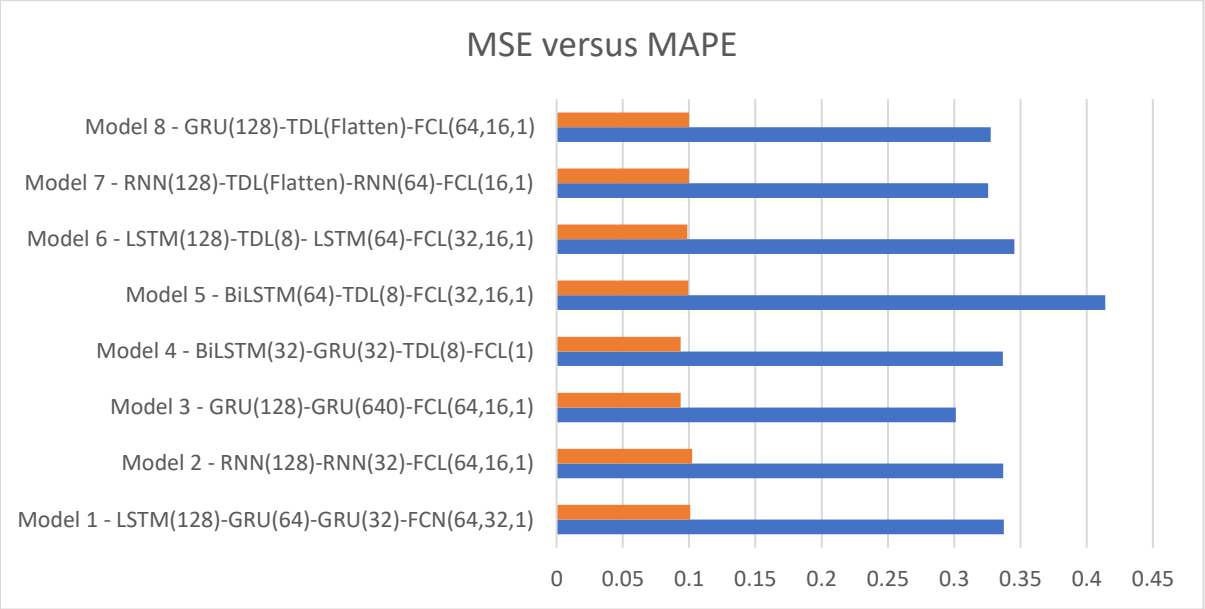


Figure 4.14 RMSE Vs MAPE comparison of different models

4.2.10 Model – Tuning – LSTM-GRU-GRU-FCL

Based on the best results, GRU-GRU-FCL model-3 was selected and further tuned to find the best hyper-parameters to get the best possible model. We ran 20 experiments to tune the nodes

in the first layer, second layer and one hidden fully connected layer. We also tuned the learning rate and activation function in the first input layer. Table 4.11 below shows the results of the different experiments.

Table 4.11 Results after tuning GRU

ID	GRU LAYER 1	GRU LAYER 2	FCL LAYER 1	ACTIVATION	LEARNING RATE	TRAIN RMSE	TEST RMSE
1	64	32	64	tanh	0.009010576	0.009694367	0.009196458
2	64	64	64	tanh	0.006873985	0.009646788	0.009164346
3	32	32	448	tanh	0.003768511	0.009743505	0.009259196
4	**128**	**128**	**32**	**relu**	**0.002365406**	**0.009536132**	**0.009069429**
5	64	128	64	relu	0.000114951	0.009967635	0.009573216
6	128	128	192	tanh	0.009287117	0.102301612	0.101421058
7	64	32	480	relu	0.005851325	0.009607912	0.009637366
8	32	64	352	tanh	0.001313116	0.009823689	0.009375519
9	128	64	448	tanh	0.005526763	0.009703883	0.009256232
10	32	32	320	relu	0.000994365	0.009826197	0.010392991
11	32	128	256	tanh	0.001420018	0.009803467	0.009325074
12	128	32	384	tanh	0.00993434	0.102301657	0.101421058
13	128	32	224	relu	0.002116337	0.009587778	0.009129008
14	128	32	288	tanh	0.004183045	0.009638422	0.009083049
15	64	32	96	tanh	0.009603613	0.009690539	0.009169068
16	128	32	160	tanh	0.008288989	0.009652188	0.009136504
17	64	32	416	relu	0.007138724	0.009621038	0.010691174
18	128	64	192	tanh	0.00806062	0.102301657	0.101421058
19	32	64	384	tanh	0.002025919	0.00975249	0.009243688
20	64	32	288	relu	0.005011946	0.009699194	0.009404447

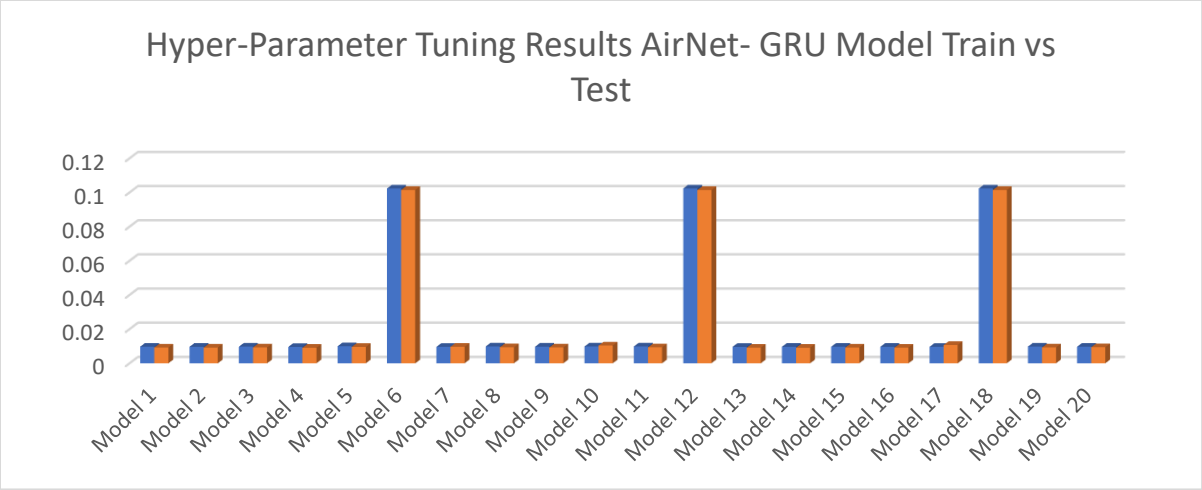


Figure 4.15 Hyper-parameter tuning of AirNet-GRU model

4.3 EPA USA – California Dataset

The EPA (Environmental Protection Agency) maintains a database with all of the data from the AQS (Air Quality Subsystem). It contains all of the measured values obtained by the EPA through the national ambient air monitoring program. It also provides the related aggregate values computed by the Environmental Protection Agency (EPA) (8-hour, daily, annual, etc.). We have considered the data for California region from 2018 to 2021. Table 4.12 below shows some of the observations for the EPA USA California Dataset.

Table 4.12 A snapshot of EPA USA California Dataset

	CO (ppm)	NO2 (ppb)	Wind (Knots)	O3 (ppm)	SO2 (ppb)	HAP (mig/m3)	PM10 (mig/m3)	RH (%)	Temperature (Fahrenheit)	Pressure (mig/m3)
DATE										
02-01-2018	0.59	22.14	1.2	0.02	0.22	0.02	33	55.42	62.38	0.02
02-01-2018	0.59	22.14	1.2	0.02	0.22	0.02	33	55.42	62.38	0.2
02-01-2018	0.59	22.14	1.2	0.02	0.22	0.02	33	55.42	62.38	0
02-01-2018	0.59	22.14	1.2	0.02	0.22	0.02	33	55.42	62.38	0.1
02-01-2018	0.59	22.14	1.2	0.02	0.22	0.02	33	55.42	62.38	4

4.3.1 Model 1 - LSTM (128)-GRU (64)-GRU (32)-FCN (64, 32, 1)

Table 4.13 shows the architecture of Model 1 with 128 Units of LSTM, 64 units of GRU followed by another layer of 32 Units of GRU followed by three layers of FCL with 64, 32, 1 Units. There are three dropout layers, one after LSTM and second and third after GRU layers. Table 4.13 shows the output shape at the respective layers and the total number of trainable parameters associated with the layers.

Table 4.13 Architecture of Model 1

Layer (type)	Output Shape	Parameters
LSTM	(None, 9, 128)	69120
Dropout	(None, 9, 128)	0
GRU	(None, 9, 64)	37248
Dropout	(None, 9, 64)	0
GRU	(None, 32)	9408
Dropout	(None, 32)	0
FCL 1 (Dense)	(None, 32)	2112
FCL 2 (Dense)	(None, 1)	33

Total Parameters: 120,001
Trainable Parameters: 120,001
Non-trainable Parameters: 0

The Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.1741 and 0.0178, respectively. This model achieved the best MAPE and the best RMSE values among all 8 models. So, this is the best model among all the models. Figure 4.16 shows the true and predicted values for some test observations. From the figure we can see that there is difference between true values and predicted values which is significantly close to each other.

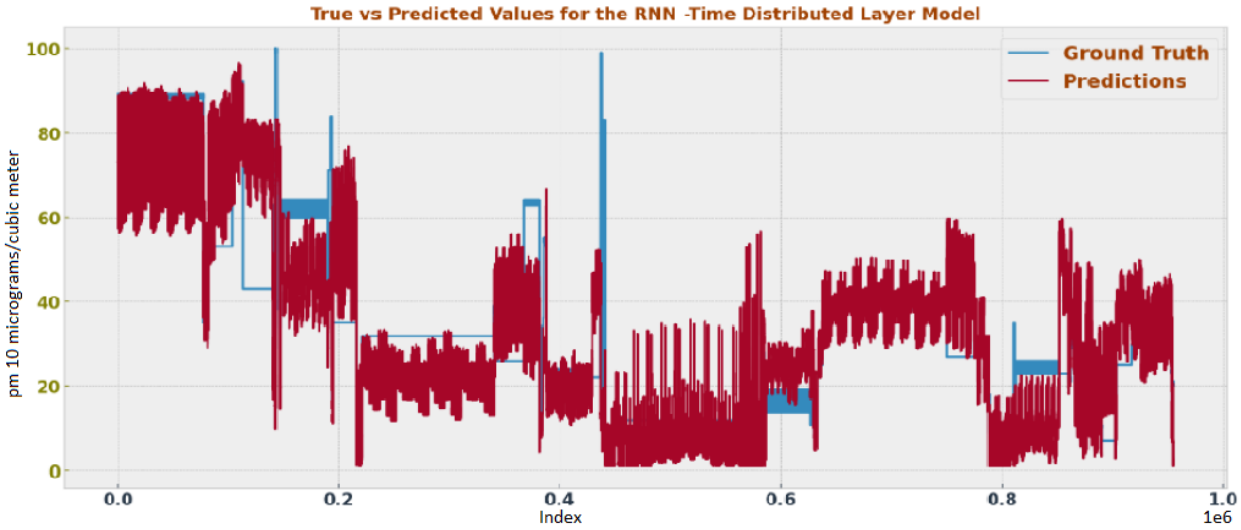


Figure 4.16 PM10 of the Ground Truth and the predicted values of Model 1

4.3.2 Model 2 - RNN (128)-RNN (32)-FCL (64, 16, 1)

Table 4.14 shows the architecture of Model 2 with 128 units of RNN followed by another layer of 32 Units of RNN, followed by three layers of FCL with 64, 16, 1 Units. There are two dropout layers, one after RNN and second after second RNN layer. Table 4.14 shows the output shape received at the respective layer and total number of trainable parameters associated with the layers.

Table 4.14 Architecture of Model 2

Layer (type)	Output Shape	Parameters
Simple RNN	(None, 9, 128)	16640
Dropout	(None, 9, 128)	0
Simple RNN	(None, 32)	5152
Dropout	(None, 32)	0
FCL 1 (Dense)	(None, 64)	2112
FCL 2 (Dense)	(None, 16)	1040
FCL 3 (Dense)	(None, 1)	17

Total Parameters: 24,961
Trainable Parameters: 24,961
Non-trainable Parameters: 0

The Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.3107 and 0.0349, respectively. This model has the worst MAPE value among all 8 models. Figure 4.17 shows the true and predicted values for some test observations. From the figure we can see that there is difference between true values and predicted values which is significant.

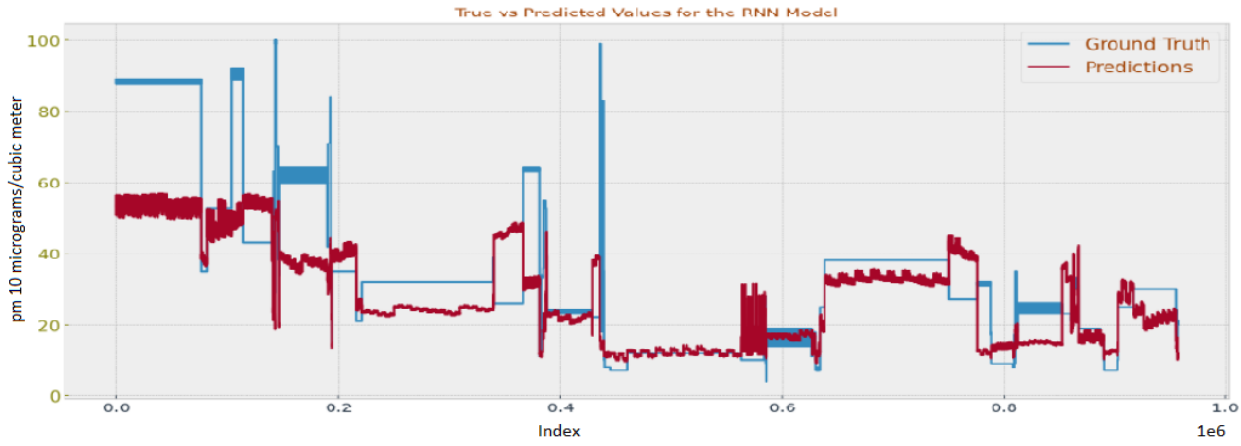


Figure 4.17 PM10 of the Ground Truth and the predicted values of Model 2

4.3.3 Model 3 - GRU (128)-GRU (64)-FCL (64, 16, 1)

Table 4.15 shows the architecture of Model 3 with 128 units of GRU layer, followed by another layer of 64 Units of GRU, followed by three layers of FCL with 64, 16, 1 Units. There are two dropout layers, one after GRU and second after second GRU layer. Table 4.15 shows the output shape received at the respective layer and total number of trainable parameters associated with the layers.

Table 4.15 Architecture of Model 3

Layer (type)	Output Shape	Parameters
GRU	(None, 9, 128)	50304
Dropout	(None, 9, 128)	0
GRU	(None, 64)	37248
Dropout	(None, 64)	0
Dense	(None, 64)	4160
FCL 1 (Dense)	(None, 16)	1040
FCL 2 (Dense)	(None, 1)	17

Total Parameters: 92,769
Trainable Parameters: 92,769
Non-trainable Parameters: 0

The Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.9964 and 0.0462, respectively. Figure 4.18 shows the true and predicted values for some test observations. From the figure we can see that there is difference between true values and predicted values which is significant.

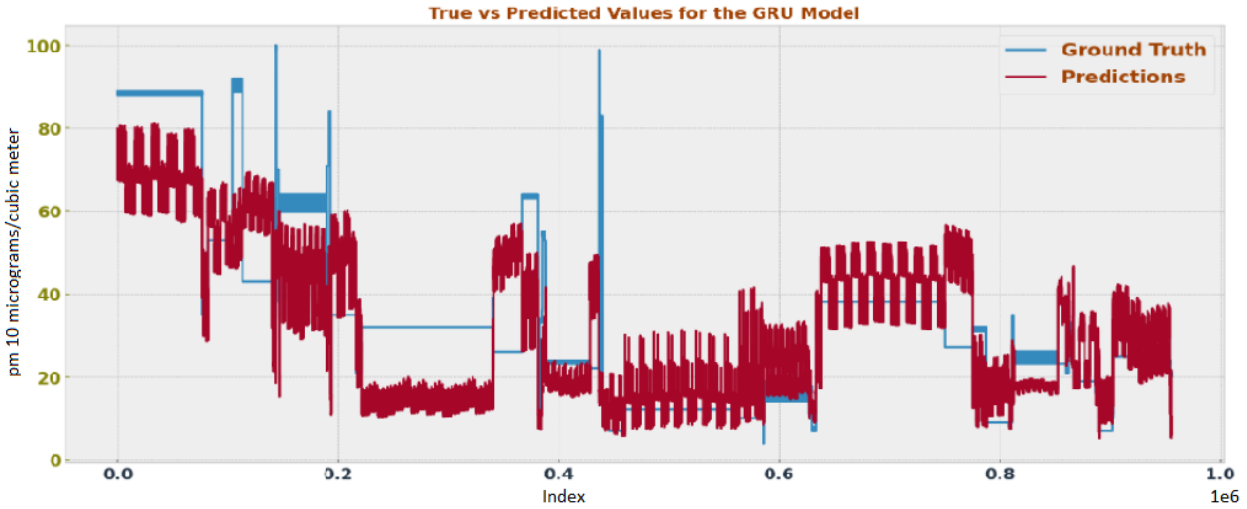


Figure 4.18 PM10 of the Ground Truth and the predicted values of Model 3

4.3.4 Model 4 - BiLSTM (32)-GRU (32)-TDL (8)-FCL (1)

Table 4.16 shows the architecture of Model 4 with 32 units of Bidirectional LSTM layer, 32 Units of GRU, 8 Units of Time Distributed Layer (TDL) followed by one layer of FCL with 1 Unit. There are two dropout layers, one after Bidirectional LSTM and second after GRU layer. Table 4.16 shows the output shape received at the respective layer and total number of trainable parameters associated with the layers.

Table 4.16 Architecture of Model 4

Layer (type)	Output Shape	Parameters
Bidirectional	multiple	8704
Dropout	multiple	0
GRU	multiple	9408
Dropout	multiple	0
Time-Distributed	multiple	264
FCL 1 (Dense)	multiple	9

```
Total Parameters: 18,385
Trainable Parameters: 18,385
Non-trainable Parameters: 0
_________________________________________________________________
```

The Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.9963 and 0.0471, respectively. Figure 4.19 shows the true and predicted values for some test observations. From the figure we can see that there is difference between true values and predicted values which is significant.

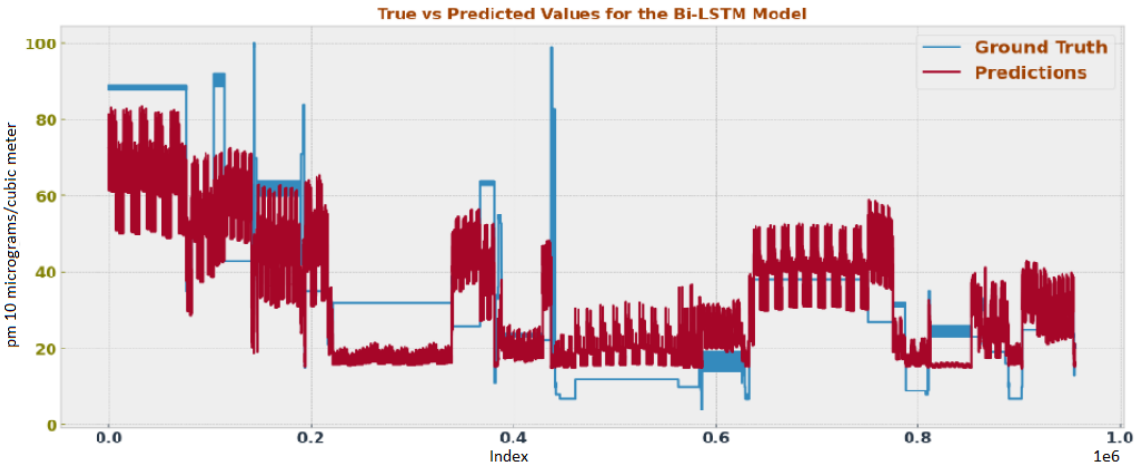


Figure 4.19 PM10 of the Ground Truth and the predicted values of Model 4

4.3.5 Model 5 - BiLSTM (64)-TDL (8)-FCL (32, 16, 1)

Table 4.17 shows the architecture of Model 5 with 64 Units of Bidirectional LSTM, 8 Units of Time Distributed Layer (TDL) followed by three layers of FCLs with 32,16 and 1 Units. There are three dropout layers, one after Bidirectional LSTM, second after first FCL layer and third right after second FCL layer. Table 4.17 shows the output shape received at the respective layer and total number of trainable parameters associated with the layers.

Table 4.17 Architecture of Model 5

Layer (type)	Output Shape	Parameters
Bidirectional	(None, 9, 128)	33792
(Dropout)	(None, 9, 128)	0
Time-Distributed	(None, 9, 8)	1032
FCL 1 (Dense)	(None, 9, 32)	288
FCL 2 (Dense)	(None, 9, 16)	528
FCL 3 (Dense)	(None, 9, 1)	17

Total Parameters: 35,657
Trainable Parameters: 35,657
Non-trainable Parameters: 0

The Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.997013 and 0.0488 respectively. This model has the worst RMSE value among all 8 models. Figure 4.20 shows the true and predicted values for some test observations. From the figure we can see that there is difference between true values and predicted values which is significant.

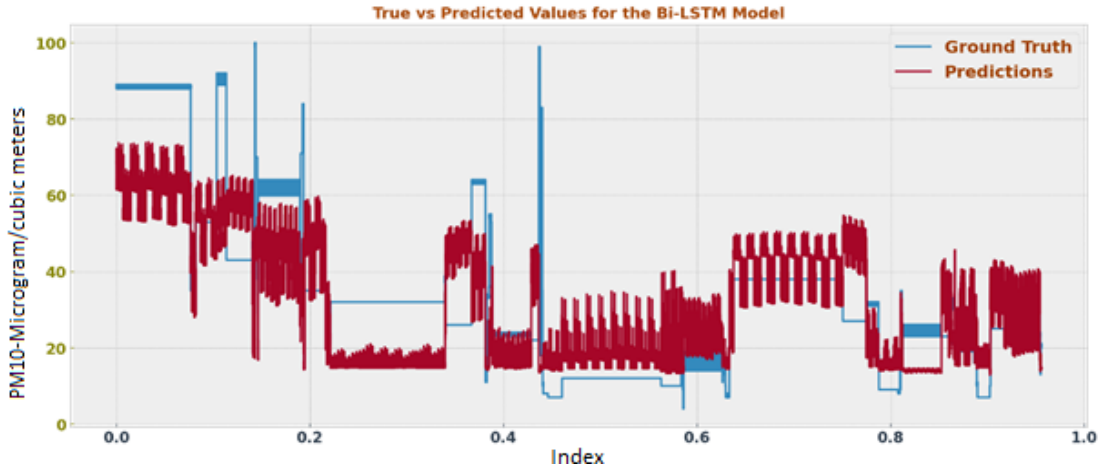


Figure 4.20 PM10 of the Ground Truth and the predicted values of Model 5

4.3.6 Model 6 - LSTM (128)-TDL (8) - LSTM (64)-FCL (32, 16, 1)

Table 4.18 shows the architecture of Model 6 with 128 Units of LSTM, 8 Units of Time Distributed Layer (TDL), 64 Units of LSTM followed by three layers of FCLs with 32,16 and 1 Units. There are four dropout layers, one after first LSTM layer, second after second LSTM layer, third after first FCL and fourth right after second FCL layer. Table 4.18 shows the output shape received at the respective layer and total number of trainable parameters associated with the layers.

Table 4.18 Architecture of Model 6

Layer (type)	Output Shape	Parameters
LSTM	(None, 9, 128)	66560
Dropout	(None, 9, 128)	0
Time Distributed	(None, 9, 8)	1032
LSTM	(None, 64)	18688
Dropout	(None, 64)	0
FCL 1 (Dense)	(None, 32)	2080
Dropout	(None, 32)	0
FCL 2 (Dense)	(None, 16)	528
Dropout	(None, 16)	0
FCL 3 (Dense)	(None, 1)	17

Total Parameters: 88,905
Trainable Parameters: 88,905
Non-trainable Parameters: 0

The Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.385572 and 0.0476, respectively. Figure 4.21 shows the true and predicted values for some test observations. From the figure we can see that there is difference between true values and predicted values which is significant.

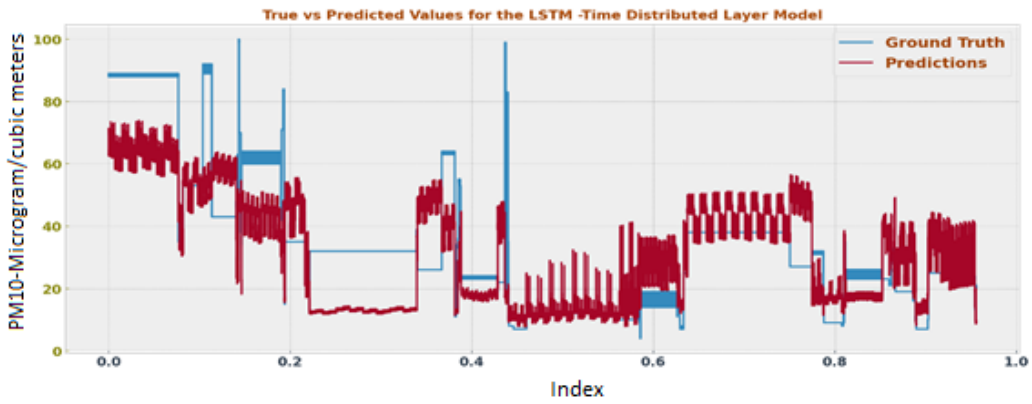


Figure 4.21 PM10 of the Ground Truth and the predicted values of Model 6

4.3.7 Model 7 - RNN (128)-TDL (Flatten)-RNN (64)-FCL (16, 1)

Table 4.19 shows the architecture of Model 7 with 128 Units of RNN, Time Distributed Layer (TDL) as a flatten layer, 64 Units of RNN followed by two layers of FCLs with 16 and 1 Units. Table 4.19 shows the output shape received at the respective layer and total number of trainable parameters associated with the layers.

Table 4.19 Architecture of Model 7

Layer (type)	Output Shape	Parameters
Simple RNN	(None, 9, 128)	16640
Time Distributed	(None, 9, 128)	0
Simple RNN	(None, 50)	8950
FCL 1 (Dense)	(None, 64)	3264
FCL 2 (Dense)	(None, 16)	1040
FCL 3 (Dense)	(None, 1)	17

Total Parameters: 29,911
Trainable Parameters: 29,911
Non-trainable Parameters: 0

The Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 0.3884 and 0.041, respectively. Figure 4.22 shows the true and predicted values for some test observations. From the figure we can see that there is difference between true values and predicted values which is not too significant.

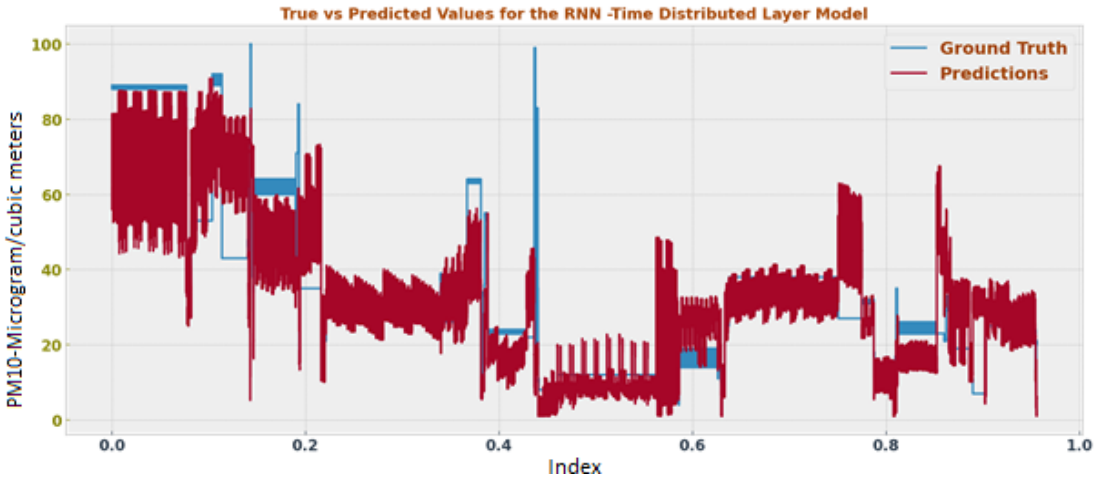


Figure 4.22 PM10 of the Ground Truth and the predicted values of Model 7

4.3.8 Model 8 - GRU (128)-TDL (Flatten) - FCL (64, 16, 1)

Table 4.20 shows the architecture of Model 8 with 128 Units of GRU, Time Distributed Layer (TDL) as a flatten layer, followed by three layers of FCLs with 64, 16 and 1 Units. Table 4.20 shows the output shape received at the respective layer and total number of trainable parameters associated with the layers.

Table 4.20 Architecture of Model 8

Layer (type)	Output Shape	Parameters
GRU	(None, 9, 128)	50304
Time Distributed	(None, 9, 128)	0
FCL 1 (Dense)	(None, 9, 64)	8256

```
_________________________________________________________________
FCL 2 (Dense)                (None, 9, 16)             1040
_________________________________________________________________
FCL 3 (Dense)                (None, 9, 1)              17
=================================================================
Total Parameters: 59,617
Trainable Parameters: 59,617
Non-trainable Parameters: 0
=================================================================
```

The Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) of the model for the test data set are 1.8705 and 0.0465, respectively. This model's MAPE value is highest among all 8 models for EPA datasets. So, we can consider this model in one of the worst models for this dataset. Figure 4.23 shows the true and predicted values for some test observations. From the figure we can see that there is difference between true values and predicted values which is not too significant.

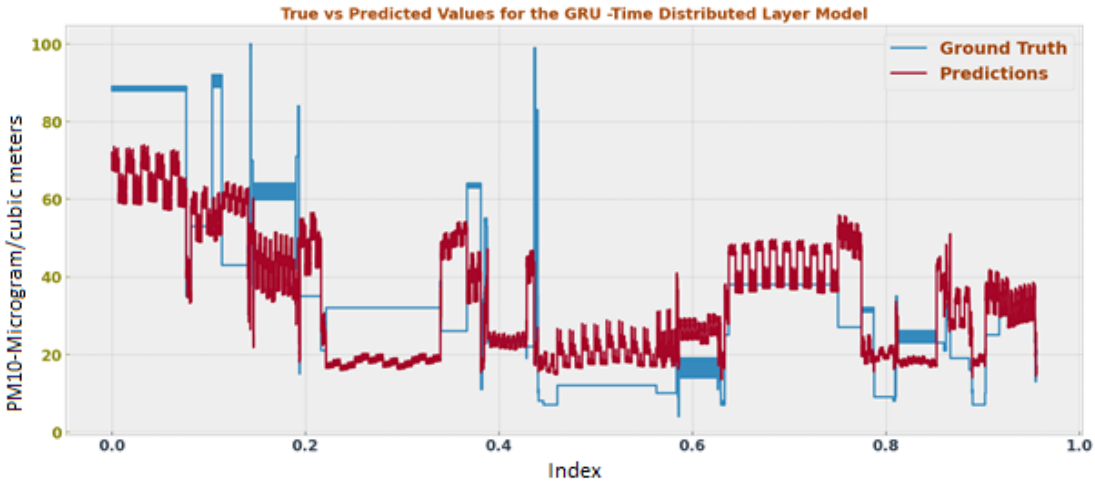


Figure 4.23 PM10 of the Ground Truth and the predicted values of Model 8

4.3.9 Comparison of Models – EPA USA Dataset

Table 4.21 and Figure 4.24-4.26 below compares all the models based on RMSE and MAPE. It can be observed that Model 1 – LSTM (128) - GRU (64) - GRU (32) – FCN (64, 32, 1) has performed the best with least value of RMSE and MAPE while the Model 8 - GRU (128) –

TDL (Flatten) – FCL (64, 16, 1) has performed the worst based on the highest value of RMSE and MAPE.

Table 4.21 Comparison of RMSE and MAPE for all the models

Model	RMSE	MAPE
Model 1 – LSTM (128)-GRU (64)-GRU (32)- FCN (64,32,1)	0.0178	0.1741
Model 2 – RNN (128)- RNN (32) - FCL (64,16,1)	0.0349	0.3107
Model 3 – GRU (128)- GRU (64) – FCL (64,16,1)	0.0462	0.9964
Model 4 – BiLSTM (32) – GRU (32) – TDL (8) – FCL (1)	0.0471	0.9963
Model 5 – BiLSTM (64)- TDL (8) - FCL (32,16,1)	0.0488	0.997013
Model 6 – LSTM (128) – TDL (8) – LSTM (64) – FCL (32,16,1)	0.0476	0.385572
Model 7 – RNN (128) - TDL (Flatten) - RNN (64) - FCL (16,1)	0.041	0.3884
Model 8 – GRU (128) – TDL (Flatten) – FCL (64,16,1)	0.0465	1.8705

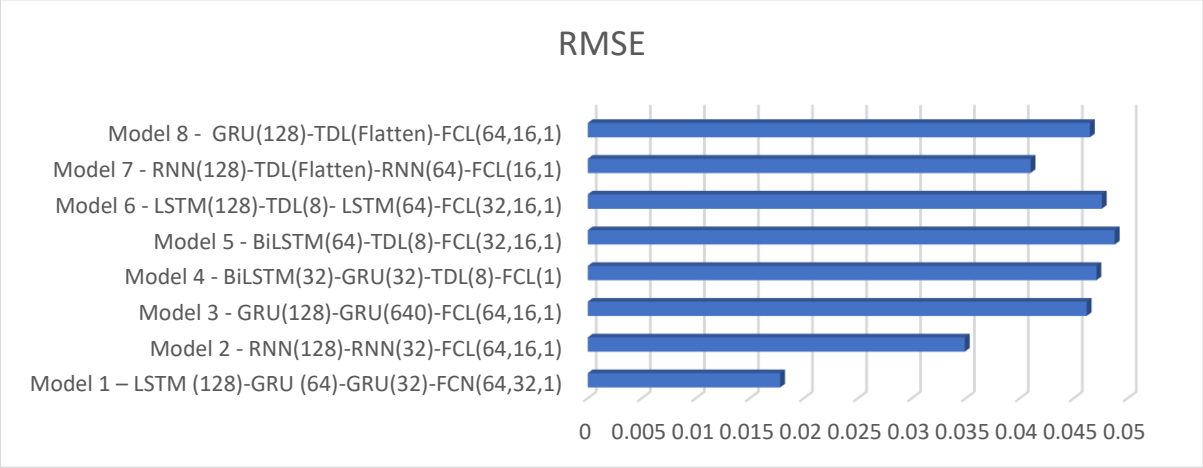


Figure 4.24 Comparison of RMSE for all the models

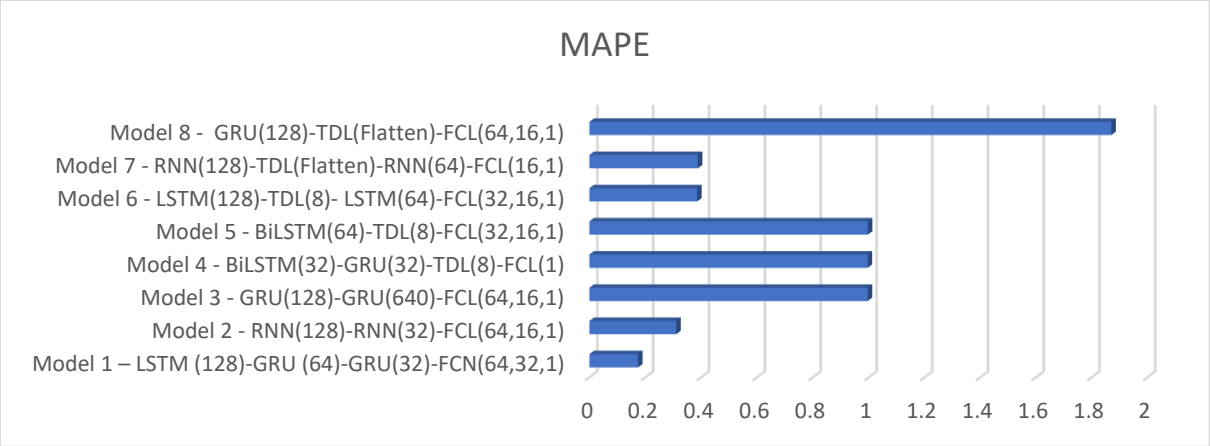


Figure 4.25 Comparison of MAPE for all the models

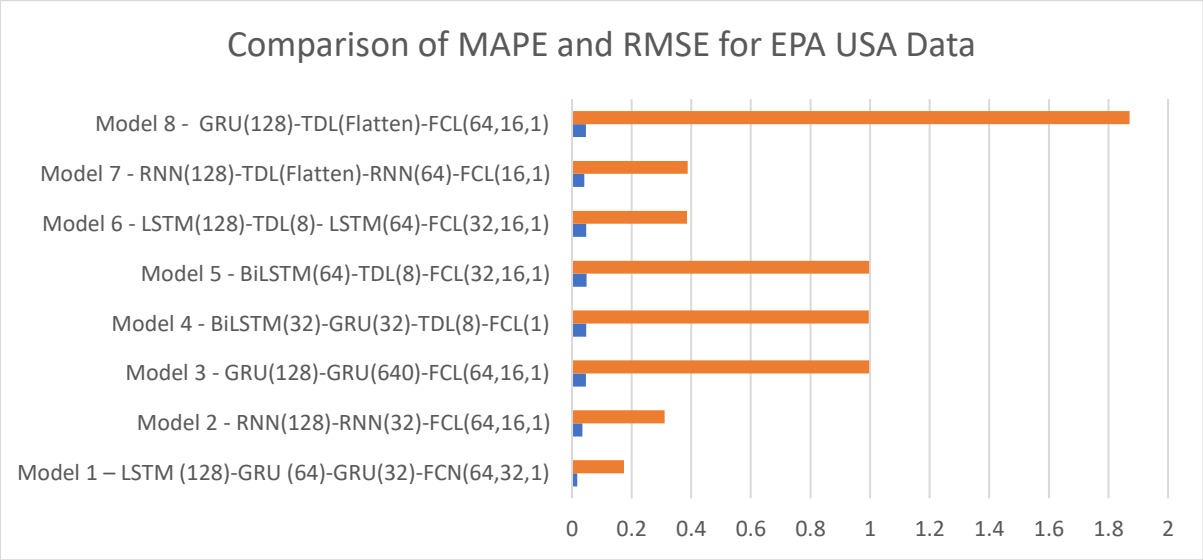


Figure 4.26 Comparison of RMSE and MAPE for all the models

4.3.10 Model - Tuning – LSTM-GRU-GRU-FCL

Based on the best results, LSTM based model was selected and further tuned to find the best hyper-parameters to get the best possible model. We ran 20 experiments to tune the node in first layer, second layer and one hidden fully connected layer. We also tuned the learning rate and

activation function in the first input layer. Table 4.22 shows the results of the different experiments and Figure 4.27 shows the bar graph for the hyperparameter tuning.

Table 4.22 Results after hyper-parameter tuning of LSTM

Model ID	LSTM Layer 1	GRU Layer 2	GRU Layer 1	FCL Layer 1	Activation	Learning Rate	Train RMSE	Test RMSE
Model 1	64	64	64	64	tanh	0.006873985	0.000387793	0.00108312
Model 2	128	128	128	192	tanh	0.009287117	0.011814344	0.01526091
Model 3	32	64	32	64	tanh	0.009010576	0.000632977	0.00189687
Model 4	32	128	32	160	tanh	0.008288989	0.000648754	0.00158841
Model 5	32	64	32	416	relu	0.007138724	0.000398056	0.00159698
Model 6	128	128	128	32	relu	0.002365406	0.000196978	0.00135499
Model 7	32	64	32	288	relu	0.005011946	0.000347323	0.00142955
Model 8	32	128	32	288	tanh	0.004183045	0.000499088	0.00153069
Model 9	64	128	64	192	tanh	0.00806062	0.000687119	0.00203039
Model 10	32	128	32	384	tanh	0.00993434	0.011814347	0.01526091
Model 11	32	128	32	224	relu	0.002116337	0.000271768	0.00141432
Model 12	64	32	64	384	tanh	0.002025919	0.000590385	0.00146421
Model 13	32	32	32	320	relu	0.000994365	0.00105168	0.00211481
Model 14	128	64	128	64	relu	0.000114951	0.001085619	0.00179563
Model 15	32	32	32	448	tanh	0.003768511	0.000620542	0.00163658
Model 16	32	64	32	480	relu	0.005851325	0.000394265	0.00174819
Model 17	128	32	128	256	tanh	0.001420018	0.000713063	0.00150746
Model 18	64	32	64	352	tanh	0.001313116	0.000860106	0.00155519
Model 19	32	64	32	96	tanh	0.009603613	0.000677942	0.00143659
Model 20	64	128	64	448	tanh	0.005526763	0.011814351	0.01526091

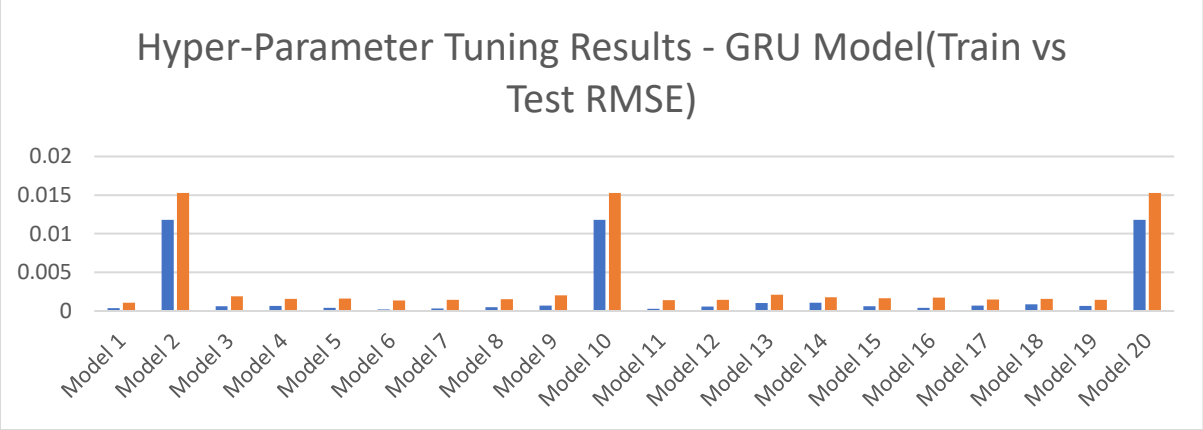


Figure 4.27 Bar graph showing the results after hyper-parameter tuning

4.4 Comparison of Results

Table 4.23 shows the comparison of all the models for the AirNet and EPA datasets and Figure 4.28 shows the bar graph. From the table we can see that Model 1 performed the best with lowest MAPE and RMSE values for EPA dataset, while Model 5 performed the worst with highest RMSE value and over an average of MAPE values. On the other end, Model 3 performed the best for AirNet Dataset while Model 5 has worst performance based on MAPE and RMSE values for the same dataset. So, we can state that if any future model with at least 2 GRU layers will perform better than any other combinations.

Table 4.23 Results of all the models for the two datasets

MODEL	ARCHITECTURE	MAPE (EPA)	RMSE (EPA)	MAPE (AirNet)	RMSE (AirNet)
Model 1	LSTM (128) – GRU (64) – GRU (32) – FCL (64,32,1)	0.1741	0.0178	0.3376	0.1008
Model 2	RNN (128) – RNN (32) – FCL (64,16,1)	0.3107	0.0349	0.337	0.1023
Model 3	GRU (128) – GRU (640) – FCL (64,16,1)	0.9964	0.0462	0.3012	0.0937
Model 4	BiLSTM (32) – GRU (32) – TDL (8) – FCL (1)	0.9963	0.0471	0.3369	0.0936
Model 5	BiLSTM (64) – TDL (8) – FCL (32,16,1)	0.997013	0.0488	0.414	0.0995
Model 6	LSTM (128) – TDL (8) - LSTM (64) – FCL (32,16,1)	0.385572	0.0476	0.3456	0.0988
Model 7	RNN (128) – TDL (Flatten) – RNN (64) – FCL (16,1)	0.388449	0.041	0.3256	0.0998
Model 8	GRU (128) – TDL (Flatten) – FCL (64,16,1)	1.8705	0.0465	0.3276	0.0999

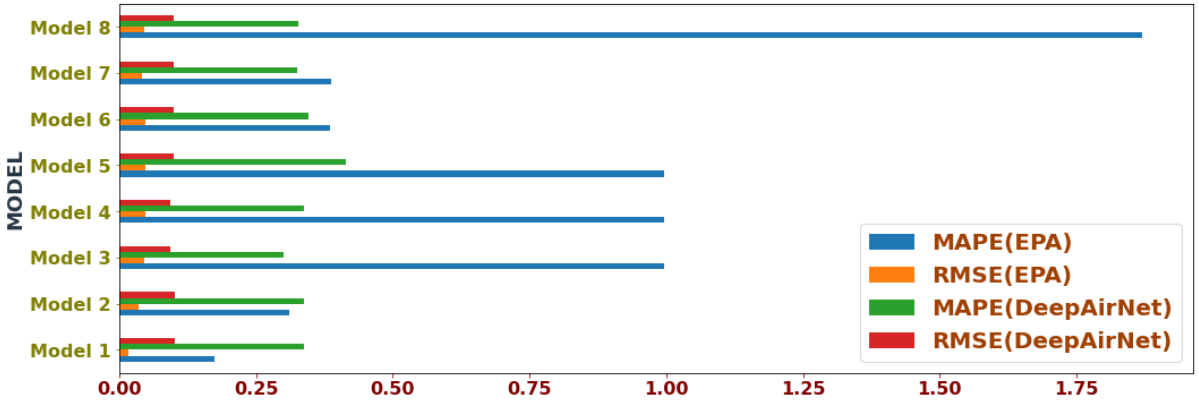


Figure 4.28 Bar graph showing the results for all the models for both the datasets

4.5 Statistical Analysis of Results

Machine learning models are mathematical and probabilistic, and they perform by finding trends in data and making predictions using simulations and statistical techniques. There's a possibility that experiments, which include taking samples from a population could reveal an impact caused by sampling errors. If the observed effect has a p-value < 0.05 (95% confidence interval (CI)), it can be concluded that the observed effect represents the characteristics of the entire population. If the p-value > 0.05, though, the observed effect does not represent the characteristics of the entire population. Statistical significance tests are used to determine if the discrepancies between the groups under study are meaningful or merely coincidental. As a result, we looked at statistical analysis of the model's results to see if there was a statistically meaningful discrepancy in the mean values of the output parameters obtained using various machine learning techniques. To assess the presence of these statistically important output variations, a one-way analysis of variance, or ANOVA, was used.

To conduct this analysis, therefore, the data must meet the following criteria: (a) normal distribution;(b) lack of major outliers. To check for data normality, QQ-plot was used. If no statistically meaningful difference is present in the mean value of the output measures for the various models under analysis, the null hypothesis (H0) that all models behave similarly is accepted. If a statistically relevant performance difference (P < 0.05) is discovered, the alternative explanation (H1) is accepted and H0 is rejected. The RMSE scores were subjected to a normality review. Based on the QQ-plot in Figure 4.29 and Figure 4.30 the sample does not have an extreme fair normal distribution, which we would not assume provided that it is just a sample. However, we can say that RMSE data is normal on both datasets.

4.5.1 Normality Check

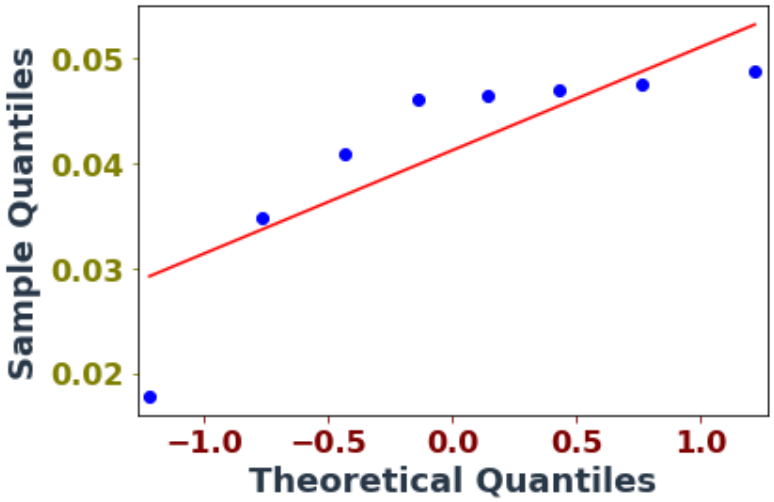


Figure 4.29 Q-Q Plot of RMSE scores of the sample data for AirNet

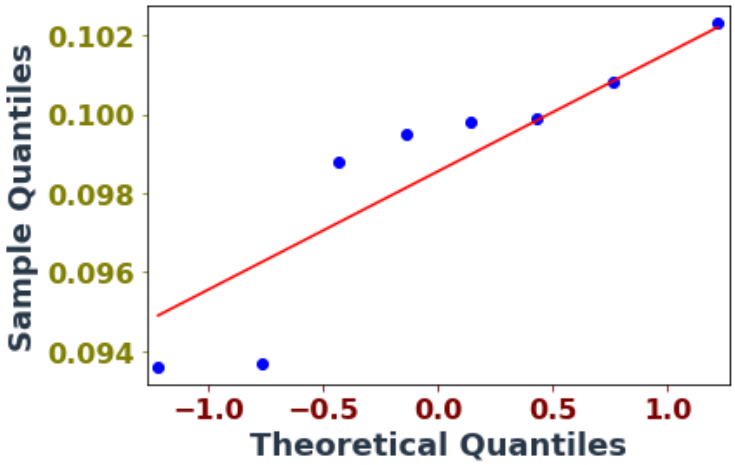


Figure 4.30 Q-Q Plot of RMSE scores of the sample data for EPA USA

4.5.2 Independent Sample t-test for the two models (AirNet and EPA USA)

Objective - The t test (also called Student's T Test) compares two averages (means) and tells if they are different from each other. The t test also tells how significant the differences are. In other words, it lets us know if those differences could have happened by chance.

A large t-score tells that the groups are different. A small t-score tells that the groups are similar. The results of the t-test are shown below.

```
T-statistics:  -14.75
D.O.F.:  14 //degrees of freedom
CV:  1.7613101357748562  //critical value
P-Value:  6.351628112355456e-10
t=-14.751, df=14, cv=1.761, p=0.000
```

The results show p-value < 0.05, which shows there is significant difference across the models and the differences are not by chance.

4.5.3 One-Way ANOVA

The result of the one-way ANOVA is shown below. We see that p-value < 0.05. Hence, we accepted the Null Hypothesis – there are differences among different density groups.

```
F-one way Result (statistic=217.5830857923553, p-value=6.351e-10)
```

4.5.4 Statistical Analysis on hyper-parameter tuning results

One way t-test

```
T-test-independent Result (statistic=-2.568, p-value=0.0142)
```

We see that p-value <0.05. Hence, we can accept the Null Hypothesis – there are differences among different density groups.

CHAPTER 5

Conclusions

5.1 Conclusions

To forecast non-linear timer series data such as weather and pollution, deep learning is increasingly gaining traction as a potential methodology in the research, we used the AirNet (China) and EPA USA (California) dataset to predict the air quality (PM10) using advanced Deep Learning Methods. For air quality prediction, a variety of deep learning models were defined. RNN, LSTM and GRU networks were used to train numerous trials to predict the PM10 values. According to the findings, the performance of the GRU network outperformed the RNN and LSTM networks for AirNet data, whereas the performance of the LSTM-based network outperformed the RNN and GRU networks for the EPA USA dataset. Also, the pattern of PM10 concentration could be identified for both datasets.

Our model will effectively predict the upcoming air quality index of any particular data inside a certain location since it is capable of predicting current data with excellent accuracy and reduced RMSE values. With this model, we can forecast the AQI and warn the appropriate regions of the country. The air quality data used in this thesis comes from China's air quality monitoring and investigation stage, and EPA USA which includes daily fine particulate matter (PM2.5), inhalable particulate matter (PM10), and total particulate matter (TPM).

5.2 Future Work

Since AirNet and EPA are large datasets, GPU resources are necessary for better performance and faster calculation. As a result, we only considered a fraction of data because of computational constraints. Even though just a fraction of the dataset was used, we were able to achieve a better forecast with low error. Convolution Neural Network (CNN) technology may be utilized to broaden the research and catch the uneven changes in air pollution data. The relationship between multiple qualities may also be assessed, giving an insight whether there is any hidden parameter that correlates the performance of features that appear to be different from the initial glance.

References

[1] Seinfeld, J. H., Pandis, S. N., 2012. Atmospheric chemistry and physics: from air pollution to climate change. John Wiley & Sons.

[2] Güler, Ü. A., & Can, Ö. P. (2017). Kimyasal Kontaminantların Çevre Sağlığı ve Gıdalar Üzerine Etkileri. Sinop Üniversitesi Fen Bilimleri Dergisi, 2(1), 170-195.

[3] Fu, M., Wang, W., Le, Z., & Khorram, M. S. (2015). Prediction of particular matter concentrations by developed feed-forward neural network with rolling mechanism and gray model. Neural Computing and Applications, 26(8), 1789-1797.

[4] Challoner, A., Pilla, F., & Gill, L. (2015). Prediction of indoor air exposure from outdoor air quality using an artificial neural network model for inner city commercial buildings. International journal of environmental research and public health, 12(12), 15233-15253.

[5] Grivas, G., & Chaloulakou, A. (2006). Artificial neural network models for prediction of PM10 hourly concentrations, in the Greater Area of Athens, Greece. Atmospheric environment, 40(7), 1216-1229.

[6] Božnar M. Lesjak M. Mlakar P. 1993A neural network-based method for short-term predictions of ambient SO2 concentrations in highly polluted industrial areas of complex terrain. Atmospheric Environment, B 27(2), 221-230.

[7] Seaton, A., Godden, D., MacNee, W., & Donaldson, K. (1995). Particulate air pollution and acute health effects. The lancet, 345(8943), 176-178.

[8] Boubel, R. W., Vallero, D., Fox, D. L., Turner, B., & Stern, A. C. (2013). Fundamentals of air pollution. Elsevier.

[9] J. Burney and V. Ramanathan, “Recent climate and air pollution impacts on Indian agriculture,” PNAS, 18-Nov-2014. [Online]. Available: https://www.pnas.org/content/111/46/16319.

[10] Bank, W. (2016). The cost of air pollution: strengthening the economic case for action. Washington: World Bank Group.

[11] Akkoyunlu, A., & Erturk, F. (2002). Evaluation of air pollution trends in Istanbul. International journal of environment and pollution, 18(4), 388-398.

[12] Theang, Ong B., Sugiura, K. and Zettsu, K. (2016). Dynamically pre-trained deep recurrent neural networks using environmental monitoring data for predicting PM2.5. Neural Compute & Applic 27:1553–1566.

[13] Kurt, A., Gulbagci, B., Karaca, F., & Alagha, O. (2008). An online air pollution forecasting system using neural networks. Environment International, 34(5), 592-598.

[14] Wan, F., & Lei, K. S. (2008). Adaptive Neuro-Fuzzy Approach to Air Pollution Prediction in Macau.

[15] Cai, M., Yin, Y., & Xie, M. (2009). Prediction of hourly air pollutant concentrations near urban arterials using artificial neural network approach. Transportation Research Part D: Transport and Environment, 14(1), 32-41.

[16] Pérez, P., Trier, A., & Reyes, J. (2000). Prediction of PM2.5 concentrations several hours in advance using neural networks in Santiago, Chile. Atmospheric Environment, 34(8), 1189-1196.

[17] Kolehmainen, M., Martikainen, H., & Ruuskanen, J. (2001). Neural networks and periodic components used in air quality forecasting. Atmospheric Environment, 35(5), 815-825.

[18] Shi, Xingjian & Chen, Zhourong & Wang, Hao & Yeung, Dit-Yan & Wong, Wai Kin & WOO, Wang-chun. (2015). Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting.

[19] Liang X, Zou T, Guo B, Li S, Zhang H, Zhang S, Huang H, Chen S. (2015) "Assessing Beijing's PM2.5pollution: severity, weather impact, APEC and winter heating." Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science 471: 20150257

[20] Liang X, Li S, Zhang S, Huang H, Chen S. (2016) "PM2.5data reliability, consistency, and air quality assessment in five Chinese cities." Journal of Geophysical Research: Atmospheres 121: 10,220-10,236

[21] U. Brunelli, V. Piazza, L. Pignato, F. Sorbello, and S. Vitabile, “Three hours ahead prevision of SO2 pollutant concentration using an Elman neural based forecaster,” Building and Environment, vol. 43, no. 3, pp. 304–314, 2008.

[22] G. Bontempi, S. Taieb, Y. Le Borgne, and D. Loshin, “Machine learning strategies for time series forecasting,” in Business Intelligence, pp. 59–73, Springer, Berlin, Germany, 2013.

[23] R. Sharda and R. B. Patil, “Neural networks as forecasting experts: an empirical test,” in Proceedings of the International Joint Conference on Neural Networks, pp. 491–494, San Diego, CA, USA, January 1990.

[24] L. A. Díaz-Robles, J. C. Ortega, J. S. Fu et al., “A hybrid ARIMA and artificial neural networks model to forecast particulate matter in urban areas: the case of Temuco, Chile,” Atmospheric Environment, vol. 42, no. 35, pp. 8331–8340, 2008.

[25] M. Cai, Y. Yin, and M. Xie, “Prediction of hourly air pollutant concentrations near urban arterials using artificial neural network approach,” Transportation Research Part D: Transport and Environment, vol. 14, no. 1, pp. 32–41, 2009.

[26] J. C. M. Pires, M. C. M. Alvim–Ferraz, M. C. Pereira, and F. G. Martins, “Prediction of PM10 concentrations through multi-gene genetic programming,” Atmospheric Pollution Research, vol. 1, no. 4, pp. 305–310, 2010.

[27] S. Tikhe Shruti, “Forecasting criteria air pollutants using data driven Approaches; an Indian case study,” IOSR Journal of Environmental Science, Toxicology and Food Technology (IOSR-JESTFT), vol. 3, no. 5, pp. 01–08, 2013, http://www.iosrjournals.org/iosr-jestft/pages/v3i5.html.

[28] M. Castelli, I. Goncalves, P. Ales, and L. Trujillo, An evolutionary system for ozone concentration forecasting, Springer Science Business Media, New York, NY, USA, 2016.

[29] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik, “Support vector regression machines,” Advances in Neural Information Processing Systems, vol. 1, pp. 155–161, 1997.

[30] K.-R. Müller, A. J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik, “Predicting time series with support vector machines,” Lecture Notes in Computer Science, vol. 1, pp. 999–1004, 1997.

[31] L. Cao, “Support vector machines experts for time series forecasting,” Neurocomputing, vol. 51, pp. 321–339, 2003.

[32] W.-C. Wang, K.-W. Chau, C.-T. Cheng, and L. Qiu, “A comparison of performance of several artificial intelligence methods for forecasting monthly discharge time series,” Journal of Hydrology, vol. 374, no. 3-4, pp. 294–306, 2009.

[33] Nikov, A., Karaca, F., Alagha, O., Kurt, A. and Hakkoymaz, H. (2005). AirPolTool: A WEB-BASED TOOL FOR ISTANBULAIR POLLUTION FORECASTING AND CONTROL, Proceedings of the Third International Symposium on Air Quality Management at Urban, Regional and Global Scales. 26-30 September 2005, Istanbul – Turkey.

[34] Ferhat Karaca, Alexander Nikov and Omar Alagha. NN-AirPol: a neural-networks-based method for airpollution evaluation and control, Int. J. Environment and Pollution, Vol. 28, Nos. 3/4, 2006.

[35] Kurt, Atakan & Oktay, Ayse. (2010). Forecasting air pollutant indicator levels with geographic models 3days in advance using neural networks. Expert Systems with Applications. 37. 7986-7992. 10.1016/j.eswa.2010.05.093.

[36] Matta, Gagan & Prachi, & Kumar, Nishant. (2011). Artificial neural network applications in air quality monitoring and management. ESSENCE – International Journal for Environmental Rehabilitation and Conservation. 2. 30-64.

[37] Cortina-Januchs, Guadalupe & Quintanilla, Joel & Vega-Corona, Antonio & Andina, Diego. (2015). Development of a model for forecasting of PM10 concentrations in Salamanca, Mexico. Atmospheric Pollution Research. 6. 10.5094/APR.2015.071.

[38] Balmes, J. R., Fine, J. M., & Sheppard, D. (1987). Symptomatic Bronchoconstriction after Short-Term Inhalation of Sulfur Dioxide1, 2. Am Rev Respir Dis, 136, 1117-112.

[39] Songgang Zhao, Xingyuan Yuan, Da Xiao, Jianyuan Zhang, Zhouyuan Li. (2018) "AirNet: A machine learning dataset for air quality forecasting"

[40] https://www.kaggle.com/epa/epa-historical-air-quality

[41] Nasr, G.E. & Badr, E. & Joun, C... (2002). Cross Entropy Error Function in Neural Networks: Forecasting Gasoline Demand. 381-384.

[42] Liu, C., Jin, Z., Gu, J., Qiu, C.: Short-term load forecasting using a long short-term memory net- work. In: 2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), pp. 1–6 (2017).

[43] Dogo, Eustace & Afolabi, Oluwatobi & Nwulu, Nnamdi & Twala, Bhekisipho & Aigbavboa, Clinton. (2018). A Comparative Analysis of Gradient Descent-Based Optimization Algorithms on Convolutional Neural Networks. 10.1109/CTEMS.2018.8769211.

[44] S. Ruder, "An overview of gradient descent optimization algorithms," 2016. Retrieved from http://arxiv.org/abs/1609.04747, 29 October 2018.

[45] Schaul, I. Antonoglou and D. Silver, "Unit Tests for Stochastic Optimization," arXiv Preprint arXiv:1312.6055, Dec 20, 2013.

[46] Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks Available from: https://www.researchgate.net/figure/Artificial-neural-network-architecture-ANN-i-h-1-h-2-h-n-o_fig1_321259051.

[47] Classification Performance Analysis of Printed Odia Alphabets using Neural Network based Approach-https://www.researchgate.net/figure/Feed-Forward-Back-propagation-NN_fig3_339177819.

[48] M. Venkatachalam, “Recurrent neural networks,” Medium, 22-Jun-2019. [Online]. Available: https://towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce.

[49] A Deep Generative Adversarial Architecture for Network-Wide Spatial-Temporal Traffic State Estimation -Available from: https://www.researchgate.net/figure/c-The-principle-of-a-basic-long-short-term-memory-neural-network-LSTM-NN_fig1_322419130.

[50] SFF Anti-Spoofer: IIIT-H Submission for Automatic Speaker Verification Spoofing and Countermeasures Challenge 2017 - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Bi-directional-Long-Short-Term-Memory-architecture-figure-adapted-from-20_fig2_319185718.

[51] “9.1. Gated Recurrent Units (GRU)¶ Colab [mxnet] Open the notebook in Colab Colab [pytorch] Open the notebook in Colab Colab [tensorflow] Open the notebook in Colab,” 9.1. Gated Recurrent Units (GRU) - Dive into Deep Learning 0.16.6 documentation. [Online]. Available: https://d2l.ai/chapter_recurrent-modern/gru.html. [Accessed: 15-Jul-2021].

[52] An Enhanced Optimization Scheme Based on Gradient Descent Methods for Machine Learning - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Comparing-proposed-scheme-with-others-for-a-three-and-b-five-layers_fig1_334610593 [accessed 15 Jul, 2021]

[53] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, “Convolutional neural networks: an overview and application in radiology,” Insights into Imaging, 22-Jun-2018. [Online]. Available: https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9/figures/7.

[54] V, Athira & Srikanth, Dr.geetha & Ravi, Vinayakumar & Kp, Soman. (2018). DeepAirNet: Applying Recurrent Networks for Air Quality Prediction. Procedia Computer Science. 132. 1394-1403. 10.1016/j.procs.2018.05.068.

[55] http://www.cnemc.cn/en/.

[56] https://www.ncdc.noaa.gov/data-access/model-data/model-datasets/global-forcast-system-gfs#:~:text=The%20Global%20Forecast%20System%20(GFS,moisture%20and%20atmospheric%20ozone%20concentration.