1. INTRODUCTION

1.1 INTRODUCTION TO PROJECT

Air pollution has been one of the major concerns for the developing countries such as India since the last few years. Air pollution has caused the most number of deaths in the near past and count keep increasing every year. Number shows that more than 660 million Indians breathe polluted air every day. Breathing polluted air can cause many diseases like lung cancer , asthma , heart diseases many more. Air quality Index is measured adopted by the Indian government to quantify air pollution.

According to the WHO (World Health Organization), air pollution is the contamination of the indoor or outdoor environment by any chemical, physical or biological agent that modified the natural characteristics of the atmosphere. Air pollution can be divided into 2 parts indoor pollution from households and outdoor pollution from vehicles and industry. Air pollution can be felt by Household combustion, motor vehicles, industrial facilities, and forest fires are common resources of air pollution. WHO data shows that almost all the global population (90%) breathes air that exceeds WHO Health guidelines . Every 9 out of 10 people lives where air quality exceeds WHO guidelines. The World Health Organization (WHO) reported that air pollution causes 4.2 million premature deaths per year in cities and rural areas around the world. Air pollution in the cities and rural areas causes some dangerous diseases like stroke, heart disease, lung cancer, and acute and chronic respiratory diseases. Around the globe around 2.6 billion, people are exposed to dangerous levels of household air pollution. This is the data from WHO.

Air pollution forecasting techniques are being rapidly advanced and measuring pollution increase. Traditional approaches use some mathematical and statistical techniques. This conventional forecasting model takes a lot of computational power to forecast the data. With recent advancements in technology, our proposed system come up with Deep Learning which is very good for solving real-time problems in various domains like computer vision, Natural Language Processing, and many more. With the help of the Deep Learning, we can obtain the best results, Deep Learning for the air pollutions forecasting of the data.

As you can see in the world Air pollution leads the third largest cause of death.

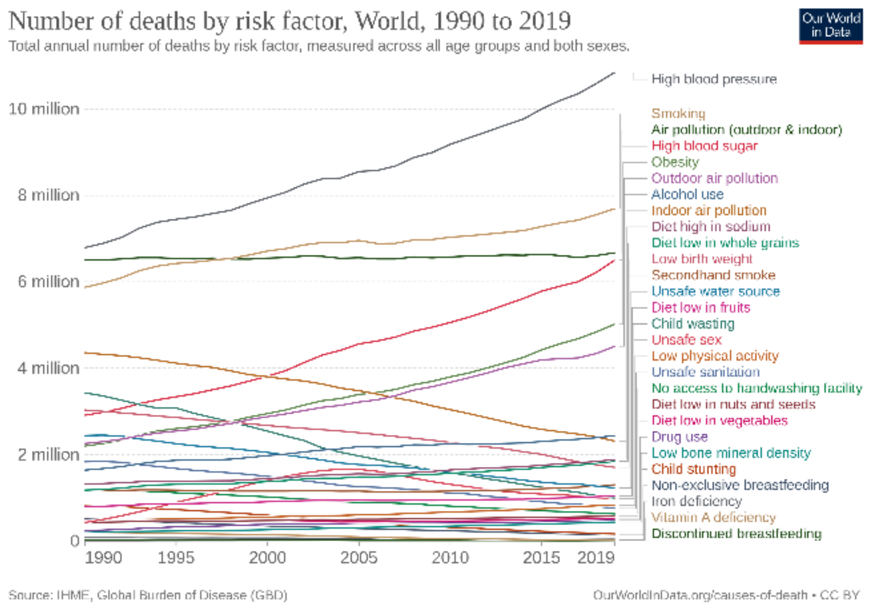


Figure 1.1 Represent the number of death cause by different diseases in the world

We can solve this problem with the help of emerging technologies like Deep Learning. Which will help us to forecast the time series data by providing the data from the previous time. In Deep Learning we have algorithms like LSTM (Long Short Term Memory) and GRU (Gated Recurrent Unit). Which will help us to forecast the data for the air pollution.

1.2 SOFTWARE AND HARDWARE SPECIFICATION

❖ Software Requirement

- Operating System : Windows 7 & above
- Tools :
    - Numpy : Numpy is used to work with the arrays and numbers.
    - Pandas : Pandas is used to understand the data and create a dataframes.
    - Tensorflow : Tensorflow is the library to create a Deep Learning model.
    - Keras : Keras is used to add the layers in Deep Learning model.
- Programming Language : Python

❖ Hardware Requirement

- Processor : Intel® Core™ i3 or Ryzen 3 above
- RAM : 4 GB or above

2. LITERATURE SURVEY

Deep Learning approaches have become more popular in last few years[1]. The most popular Deep Learning techniques are Multi-Layer Perceptron (MLP), Deep Belief Network (DBN), Convolutional Neural Networks (CNN), Recurrent Neural Network (RNN), and Auto Encoder (AE)[4].We will use Recurrent Neural network for our work to obtain the result.

Many researchers are working on this problem of air pollution forecasting nowadays. Mostly they are focusing on the LSTM (Long short term memory) or GRU (Gated Recurrent Unit). But in this research, we are going to combine both of the famous models of Recurrent Networks[4]. In many fields, these 2 models are giving their best to provide the solution to the problem.

The LSTM model can work more efficient the on the hourly basis concentration of the air pollutions[1]. In this studies they have worked with the real time data of Vishakhapatnam for the 12 hours of air pollutions[1]. Based on this data we can say that we can use the data LSTM for the air pollution forecasting.

In this studies[5] they have used the three neural network model which are (1) Multilayer Perceptron Layer (MLP) (2) Radial Basis Function (RBF) (3) Square Multilayer Perceptron (SMLP), among all this network model the RBF has performed well while executions. Combination of neural network model with the traditional Machine Learning algorithm also shows that the combination of various model can can improve the accuracy.

We can collect the data from the wireless sensors and give it to the trained LSTM model for real time prediction of data for the air pollutions[2]. Collect the real time data for some days and create model based on that collected data.

The effects of air pollution are receiving increasing attention. Many studies use a variety of machine learning approaches to build pollutant concentration prediction models or air quality predictions to predict air quality[6]. Among the most promising solutions to solve these problems are traditional templates based on machine learning[38] and deep learning techniques such as: GBTR, SVR, LSTM[6]. Run tests and compare the results with the results of the GBTR, SVR, LSTM[41], and LSTM2 (version 2) models. Experimental results show that the hybrid model is superior to the methods currently used to predict air pollution[6].

We look at a novel hybrid early warning system for air quality that includes characteristic estimation, prediction, and assessment[7]. For analyzing the properties of contaminants, two estimating approaches and four distinct distribution functions were used. Then, a hybrid forecasting model was suggested in conjunction with two cutting-edge data processing methods—a neural network and a novel heuristic algorithm[7]. Two interval techniques and Extenics assessment were used as essential elements of the created system to further mine the properties of contaminants[7].

Deep learning, neural networks, and conventional machine learning are among the study fields. We suggest a Long Short-Term Memory (LSTM) model called the Aggregated LSTM (ALSTM)[40] based on the LSTM deep learning technique[8]. We combine three LSTM models into a predictive model for early forecasts based on information from adjacent industrial air quality stations and other sources of pollution in order to increase prediction accuracy[8]. We tested our novel ALSTM model in comparison to SVR (Support Vector Machine based Regression), GBTR (Gradient Boosted Tree Regression), LSTM, etc., in the prediction of PM2.5 over 1–8 hours[8], and we assessed them using a variety of evaluation approaches, including MAE, RMSE, and MAPE[8]. The outcomes show that the suggested aggregated model may significantly increase prediction accuracy[8].

We can use the new hybrid intelligence model based on the long short term memory (LSTM) and multi-verse optimization algorithm (MVO) has been developed to predict and analysis the air-pollution obtained from combined cycle power plant[9].The non-linear , dynamic and complex character of air pollution has resulted in a lack of consistency in the predicted accuracy of majority of traditional forecasting methods. Because of their unique characteristics , such as organic learning , high precision , superior generalization , strong fault tolerance[10].

The use of traditional variational autoencoder (VAE) and the attention mechanism to develop the forecasting modeling strategy based on the innovative integrated multiple direct attention (IMDA) deep learning architecture. A result obtained finally demonstrates the satisfying performance of integrated multiple attention variation autoencoder (IMDA-VAE)[11]. Predicting based on the Recurrent Neural Network(RNN) model with along of Long short term memory (LSTM). Compare different deep learning model performance to an Auto-regressive Integrated Moving Average (ARIMA) model[12].

To assess the stability and robustness of this method, three cities in China with different economic characteristics were used. The results show that point predictions and 80% interval predictions are superior to other benchmark models[13]. LUR is more commonly used in NOx studies, but the Artificial Neural Network (ANN) approach has been adopted in PM and O3 studies. In addition, the multi-method hybrid approach has rapidly become widespread between 2010 and 2018. Interactions between pollutants will be the most difficult factor in future air pollution prediction studies if statistical air pollution predictions are expected to be based on mixed techniques that simultaneously predict large numbers of pollutants[14].

Beijing's NO2 Concentration Composite Weight Prediction Model (CWFM) is built on three separate prediction models. To make the input data more dimensional, the data is first analyzed using the discrete wavelet transform. Wavelet decomposition is then used to build DWT-LSTM, DWT-GRU, and DWT-Bi-LSTM models. The result shows that combine weight forecasting model by weight assignment[15]. This article describes a new pollution prediction model (MO-TCNA), a multi-output time convolutional network autoencoder. To perform multi-step long-term predictions of multiple pollutants and different locations in a single training model, the model accumulates predictions for each step[16].

The predictive capabilities of the gray model are enhanced by non-uniform accumulation to reduce variance information loss. The proposed model shows excellent predictive performance and algorithm efficiency compared to other models. This model is used to predict general concerns about air pollution in three cities in China and demonstrates the practical importance of the model[17].Predictive models were developed using artificial neural networks (ANN), inter-sequence long-term memory networks (LSTM), and time-series mutual validation (LSTM-CVT). The model with the best performance in each prediction interval used the proposed LSTM-CVT method with promising results[18].

Based on CNN-LSTM[30][31], we have developed a deep learning method to predict hourly concentrations in Beijing, China. We investigated how well deep learning algorithms such as LSTM[32], Bi-LSTM, GRU[39], Bi-GRU, CNN, and hybrid CNN-LSTM models work[19]. Infinite Impulse Response Filters (IIR)[33], Multilayer Perceptrons (MLP), Radial Basis Function Networks (RBF)[36], Extreme Learning Machines (ELM)[34], Echo State Networks (ESN)[35], and Adaptive Network Fuzzy Inference System are among the models that have so far been taken into consideration[20].

The paper which has reviewed is to summarize the theoretical analysis model of air pollution affecting health costs, and further explore the impact of air pollution[21][29]. The air pollution in Poland with the focus on the raspatory diseases , including asthma and many more[22]. The air pollution affecting in China find 10 percent increase in pollution[23]. The people who are migrating from one place to another even they are also facing the problem of air pollution frequently[24]. While stuck in the traffic all the vehicle are on and on that point the air pollution must be high which is Traffic-related air pollution(TRAP)[25][27][28]. The three major cities of India Maharashtra , Kolkata and Delhi air pollution based on the pre-Diwali and post-Diwali festival[26].

3. SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In the current scenario if we have to see for the air pollution then we can find the real time data means the data what the sensors has collected from the atmosphere. From that we can see the air quality index. There are many weather forecasting API but in that they are not providing the air quality forecasting of data. They will give us the information like temperature forecasting , rain forecasting , snow forecasting etc.. but they are not including the air pollution forecasting. The air pollution data are available on the government website and we can see that but that is real time what is going on currently in the atmosphere. The existing system is not capable of the air pollution forecasting of data.

3.2 LIMITATION OF THE EXISTING SYSTEM

If someone has to see the forecasting of data there is no any application which can provide the data for the air pollution. Which can lead to the problems like the breathing of polluted air.

3.3 PROPOSED SYSTEM

In the proposed system we are developing the system which can forecast the air pollution using the famous Deep Learning technique Recurrent Neural Network which are LSTM and GRU. Based on these 2 algorithms we are forecasting the data. For that we have to provide the data for the forecasting to that model. We are giving last 15 days on the hourly basis to this algorithm and based on that it will forecast the next 24 hours of data for the air pollution. We have used that algorithm both LSTM and GRU for doing this task. Even we have implemented the combine model which are LSTM + GRU for the forecasting of data. The model is quite accurate while forecasting of the data.

3.4 ADVANTAGES OF PROPOSED SYSTEM

- Provide the better accuracy.
- It will generate the next 24 hours of forecast for the air pollution.
- Provide the last 15 days of data and it will forecast the next 4 hours of data.

4. DATASET

The dataset what we have used for this project is taken from the Kaggle (Link). The dataset is for air pollutions based on the US embassy in Beijing, China. The data has collected every hour for the five years. The dataset is starting from 02-01-2010 00:00 to 31-12-2014 23:00 in total it is of 4 years in total.

The columns include in the dataset are:

(i) Date: The first column is of date & time which shows at what date and time the data has collected.

(ii) Pollution: The second column is of PM2.5 concentration which is air pollution the data what we have to forecast.

We have a range of Pollution values in the dataset which are :

Range (Pollution)	Count (Pollution)
100 to 200	4509
< 100	12959
200 to 300	1759
300 to 400	493
400 to 500	222
>800	3
500 to 600	49
700 to 800	3
600 to 700	3

Table-4.1 : Range of data and count for the Pollution Column

Max	994
Min	0
Average	94.06506
SD	92.25023

Table-4.2 : Min, Max, Average, SD for the Pollution column

The data Visualization for that column is displayed below :

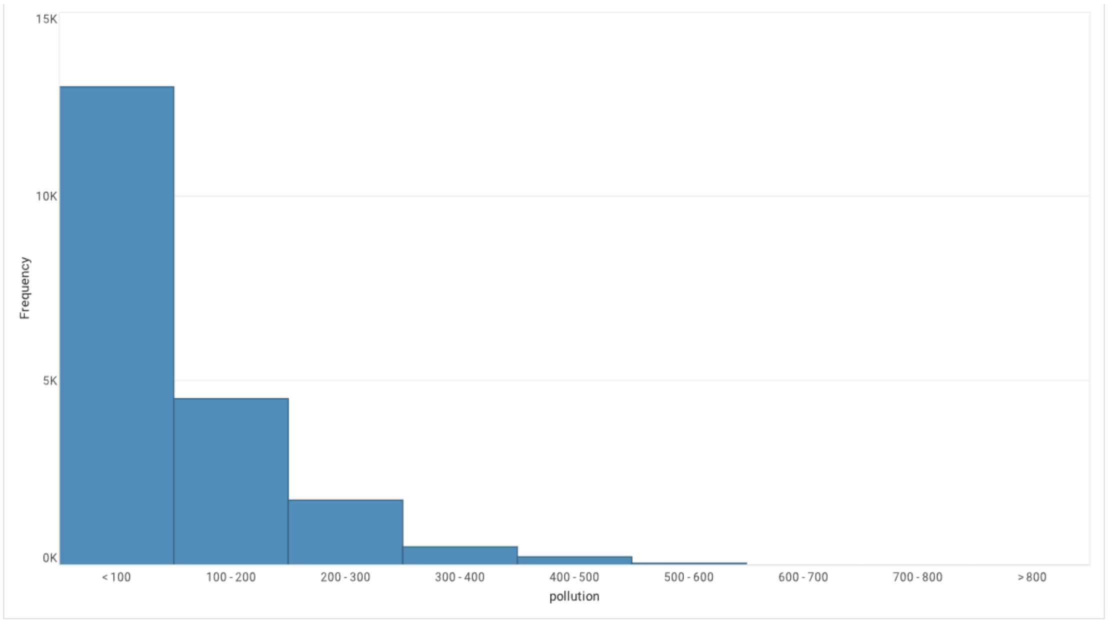


Figure 4.1 Data Visualization for Pollution column

(iii) Dew : The third column is of dew point which is the atmospheric temperature below which water droplets begin to condense and dew can form.

We have a range of values for the dew in the dataset which are :

Range (Dew)	Count (Dew)
-20 to -15	2119
-15 to -10	2155
-10 to -5	2388
-25 to -20	1331
< -25	143
-5 to 0	2183
0 to 5	1647
5 to 10	1563
10 to 15	1751
15 to 20	2313
>20	2407

Table-4.3 : Range of data and count for the Dew Column

Max	28
Min	-40
Average	1.840004
SD	14.42474

Table-4.4 : Min, Max, Average, SD for the Dew Column

The data visualization for the Dew Column look like this :

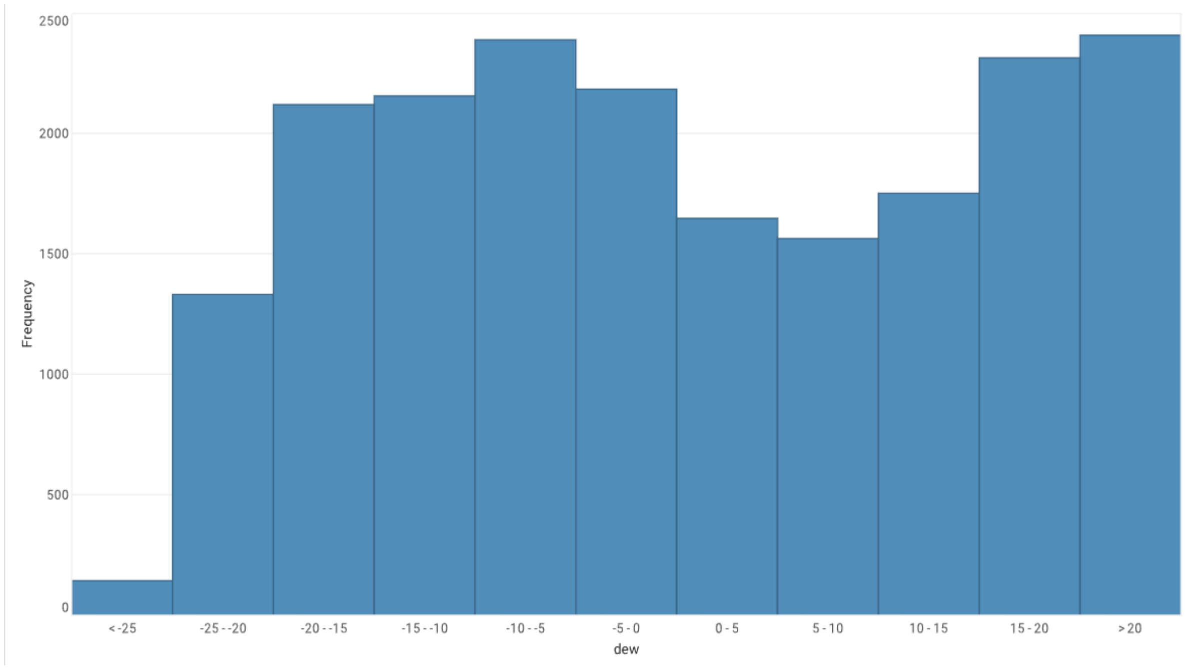


Figure 4.2 Data Visualization for the Dew Column

(iv) Temp : The fourth column is of temp which shows at that time what is the temperature at that time.

We have a range of data for the temp in the dataset which are :

Range (Temp)	Count (Temp)
-10 to 0	4335
< -10	451
0 to 10	4743
10 to 20	4205
20 to 30	5216
>30	1050

Table-4.5 : Range of data and count for the Temp Column

Max	42
Min	-19
Average	12.46955
SD	12.18802

Table-4.6 : Min, Max, Average, SD for the temp Column

The data visualization for the Temp column look like this :

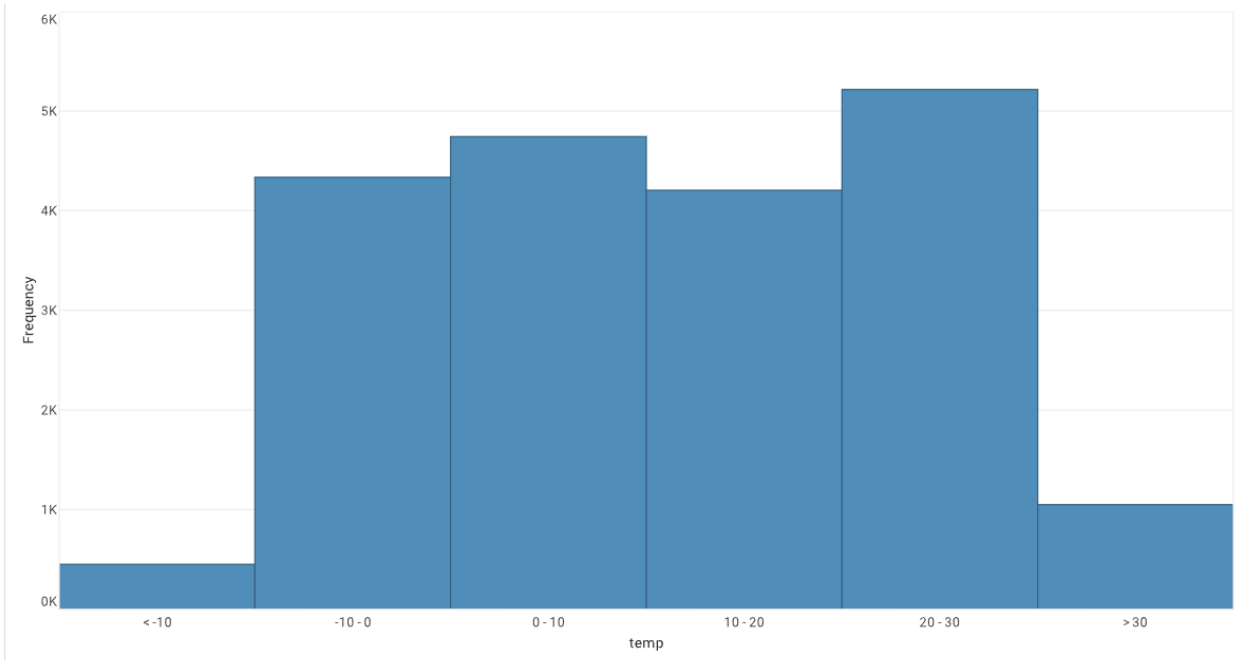


Figure 2.3 Data Visualization for the temp column

(v) Press : The fifth column is of press which is pressure the force exerted on a surface by the air above it as gravity pulls it to Earth.

We have range of values in the Press in the dataset which are :

Range (Press)	Count (Press)
1020 to 1025	3059
1025 to 1030	3110
1030 to 1035	1794
1035 to 1040	901
1015 to 1020	2774
1011 to 1015	3204
1005 to 1010	2515
1000 to 1005	1977
>1040	171
995 to 1000	488
< 995	7

Table-4.7 : Range of data and count for the Press Column

Max	1046
Min	991
Average	1016.437
SD	10.26512

Table-4.8 : Max, Min, Average, SD for the Press Column

The data visualization for the Press column look like:

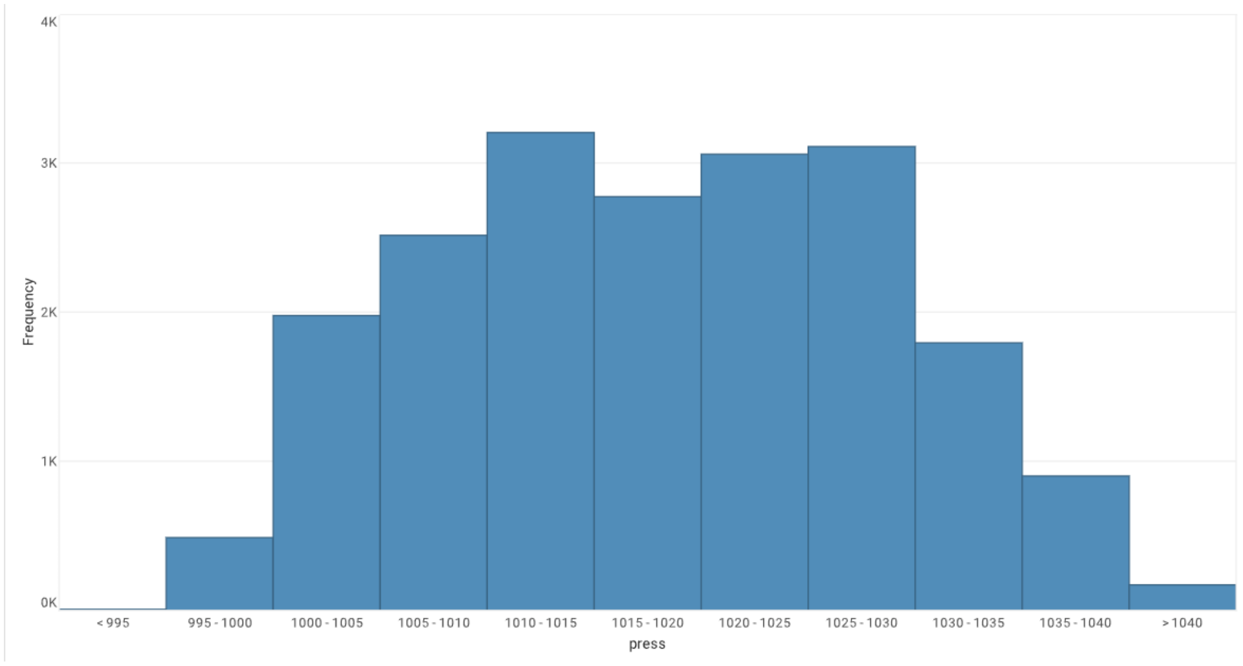


Figure 4.4 Data Visualization for Press Column

(vi) Wnd_dir : The sixth column is of wind direction, the wind direction is of combined wind direction in which direction the air is flowing.

We have a range of values in the Wind_dir in the dataset which are :

Wind Direction	Count
SE	15290
NW	14107
NE	4995
cv	9384

Table-4.9 : Count of direction in wind direction column

(vii) Wnd_spd : The seventh column is of wind speed, the wind speed is of cumulative wind speed for the entire hour.

We have a range of values in the Wind_dir in the dataset which are :

Range (Wind_spd)	Count (Wind_spd)
< 50	17034
50 to 100	1457
100 to 150	636
150 to 200	393
200 to 250	199
250 to 300	152

300 to 350	61
>350	68

Table-4.10 : Range of data and count for the Wind Speed Column

Max	565.49
Min	0.45
Average	23.82551
SD	49.68173

Table-4.11 : Max, Min, Average and SD for the wind speed Column

The data Visualization for the Wnd_spd column look like this :

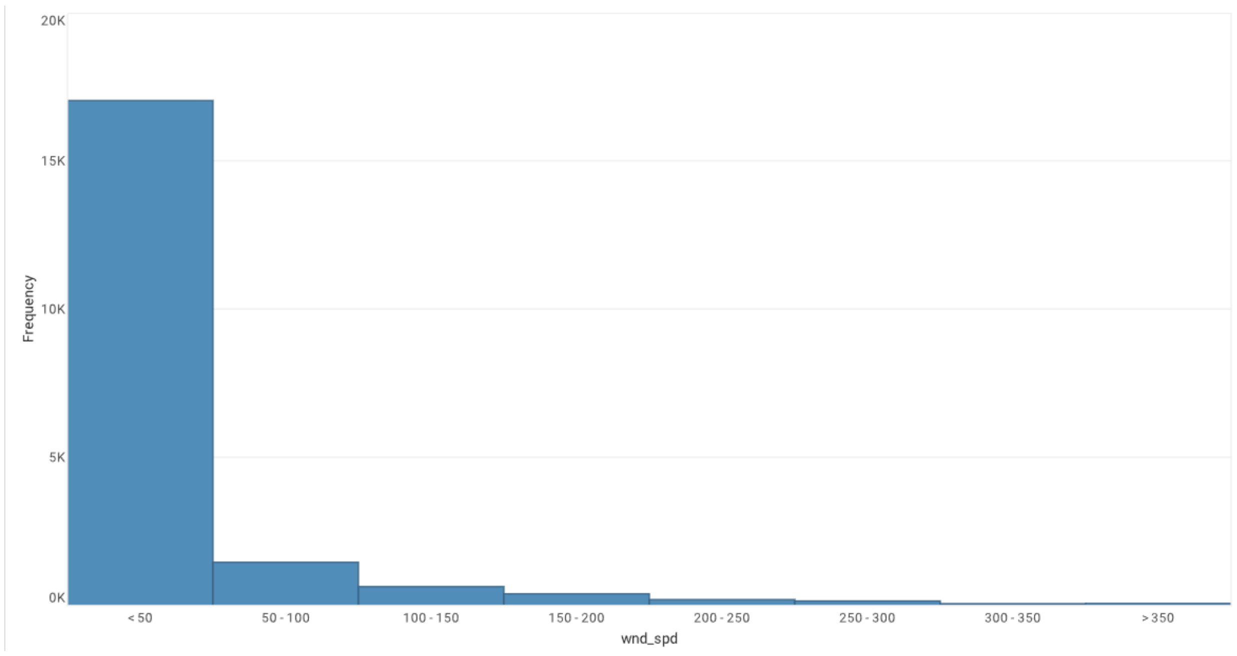


Figure 4.5 Data Visualization for the wnd_spd

(viii) Snow : The eight column is of snow, the cumulative hours of snow for the hour.

We have range of values in the snow in the dataset which are :

Range (snow)	Count (snow)
< 2.5	19848
2.5 to 5	38
5 to 7.5	41
7.5 to 10	21
10 to 12.5	21
12.5 to 15	9
15 to 17.5	9

17.5 to 20	4
20 to 22.5	4
22.5 to 25	2
>25	3

Table-4.12 : Range of data and count for the Snow Column

Max	27
Min	0
Average	0.052791
SD	0.760781

Table-4.13 : Max, Min, Average and SD for the snow Column

The data visualization for the Snow will look like this :

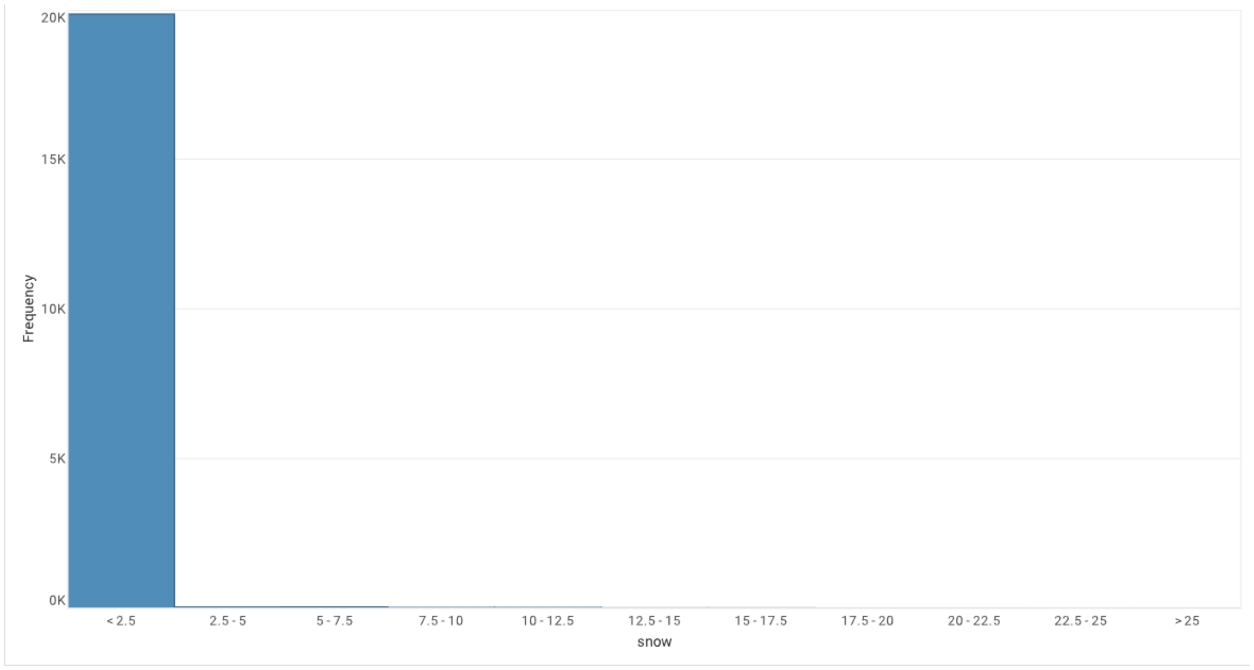


Figure 4.6 Data Visualization for the snow

(ix) Rain : The ninth column is of rain, the cumulative hours of rain for the hour.

We have range of values in the Rain in the dataset which are :

Range (Rain)	Count (Rain)
< 2.5	19587
2.5 to 5	156
5 to 7.5	119
7.5 to 10	37
10 to 12.5	40
12.5 to 15	17
15 to 17.5	13

17.5 to 20	7
>20	24

Table-4.14 : Range of data and count for the Rain Column

Max	36
Min	0
Average	0.19513
SD	1.416612

Table-4.15 : Max, Min, Average and SD for the Rain Column

The data Visualization for the Rain look like this :

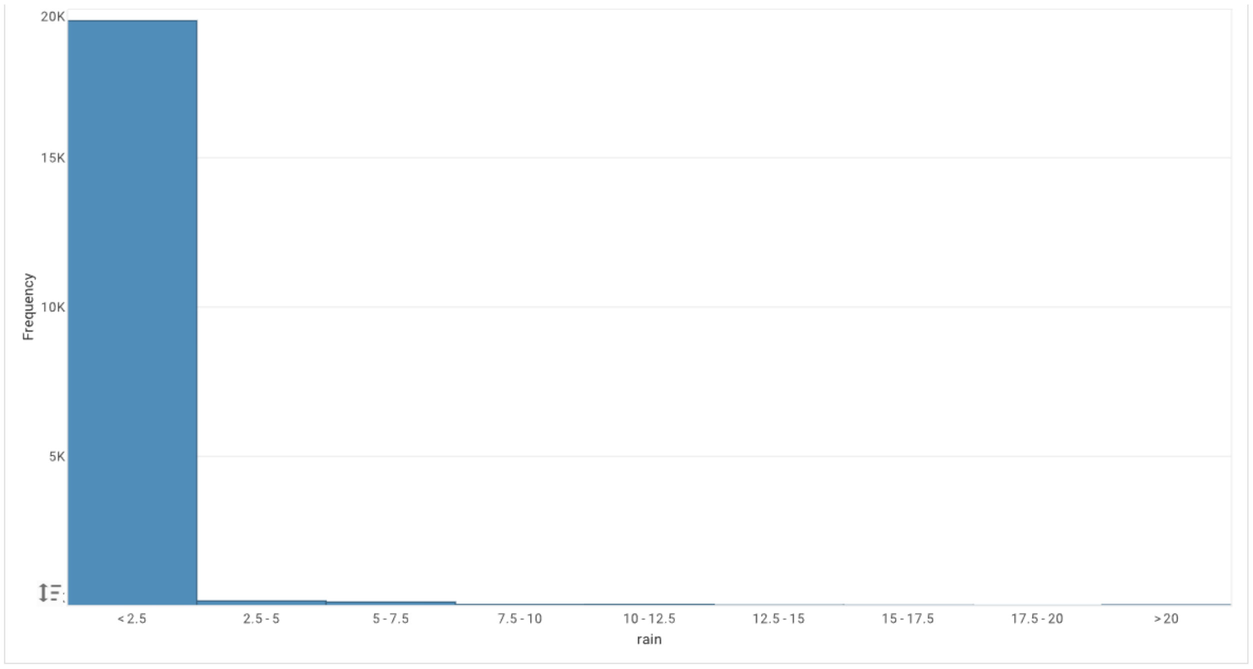
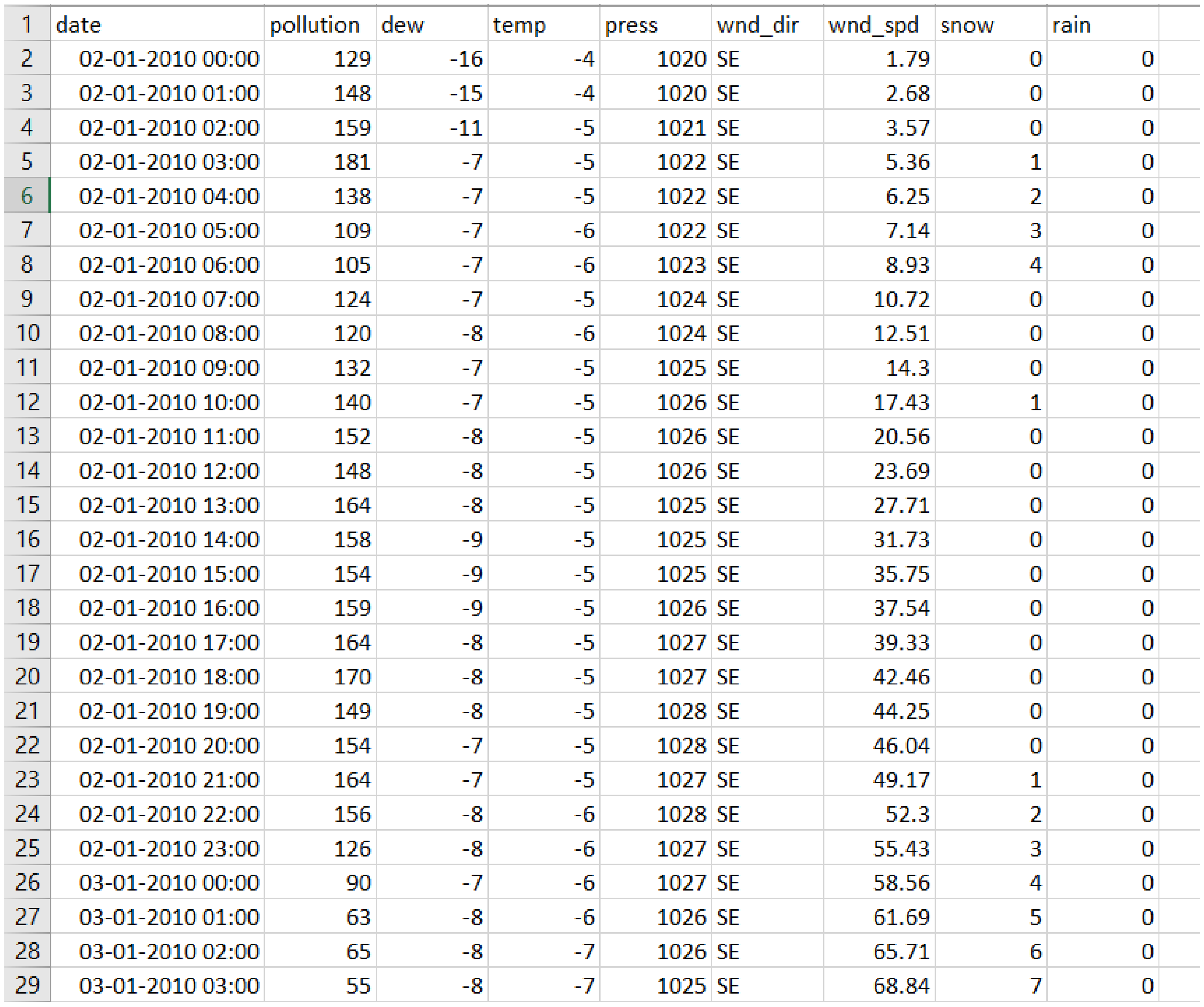


Figure 4.7 Data Visualization for the Rain



1	date	pollution	dew	temp	press	wnd_dir	wnd_spd	snow	rain
2	02-01-2010 00:00	129	-16	-4	1020	SE	1.79	0	0
3	02-01-2010 01:00	148	-15	-4	1020	SE	2.68	0	0
4	02-01-2010 02:00	159	-11	-5	1021	SE	3.57	0	0
5	02-01-2010 03:00	181	-7	-5	1022	SE	5.36	1	0
6	02-01-2010 04:00	138	-7	-5	1022	SE	6.25	2	0
7	02-01-2010 05:00	109	-7	-6	1022	SE	7.14	3	0
8	02-01-2010 06:00	105	-7	-6	1023	SE	8.93	4	0
9	02-01-2010 07:00	124	-7	-5	1024	SE	10.72	0	0
10	02-01-2010 08:00	120	-8	-6	1024	SE	12.51	0	0
11	02-01-2010 09:00	132	-7	-5	1025	SE	14.3	0	0
12	02-01-2010 10:00	140	-7	-5	1026	SE	17.43	1	0
13	02-01-2010 11:00	152	-8	-5	1026	SE	20.56	0	0
14	02-01-2010 12:00	148	-8	-5	1026	SE	23.69	0	0
15	02-01-2010 13:00	164	-8	-5	1025	SE	27.71	0	0
16	02-01-2010 14:00	158	-9	-5	1025	SE	31.73	0	0
17	02-01-2010 15:00	154	-9	-5	1025	SE	35.75	0	0
18	02-01-2010 16:00	159	-9	-5	1026	SE	37.54	0	0
19	02-01-2010 17:00	164	-8	-5	1027	SE	39.33	0	0
20	02-01-2010 18:00	170	-8	-5	1027	SE	42.46	0	0
21	02-01-2010 19:00	149	-8	-5	1028	SE	44.25	0	0
22	02-01-2010 20:00	154	-7	-5	1028	SE	46.04	0	0
23	02-01-2010 21:00	164	-7	-5	1027	SE	49.17	1	0
24	02-01-2010 22:00	156	-8	-6	1028	SE	52.3	2	0
25	02-01-2010 23:00	126	-8	-6	1027	SE	55.43	3	0
26	03-01-2010 00:00	90	-7	-6	1027	SE	58.56	4	0
27	03-01-2010 01:00	63	-8	-6	1026	SE	61.69	5	0
28	03-01-2010 02:00	65	-8	-7	1026	SE	65.71	6	0
29	03-01-2010 03:00	55	-8	-7	1025	SE	68.84	7	0

Figure 4.8 The sample data for the dataset which has collected

5. SYSTEM DESIGN AND DEVELOPMENT

5.1 METHODOLOGY

Our proposed use the most popular framework of Deep Learning which is LSTM (Long Short term memory) and GRU (Gated Recurrent Unit). As everyone know LSTM which has the special ability for storing the previous execution data and store in the memory and can be used for predicting data. Recurrent Neural networks is not efficient when it comes to the long sequence of data. To overcome that Deep Learning have introduce the LSTM(Long Short Term Memory) and GRU(Gated Recurrent Unit) which are useful when we have a long sequence of data.

LSTM and GRU were introduce to overcome the problem of long sequence of data. What it do is that it will they have internal mechanism of memory cell (The combination of gates is called memory cell). When the data are provided to the memory cell then it will define that whether to use that data or remove the data. The LSTM has a similar control flow as the recurrent neural network. It will send the data back again to it only while doing forward propagation. Every LSTM memory cell has the following Sigmoid and Tanh Activation function and three gates which are Input Gate, Forget Gate, and Output Gate.

If you see in the dataset the column with the name wnd_dir which indicate the wind direction in which direction the wind is flowing. The data are represented in the text values with the categorical data. And we cannot give that kind of data to the model because it will not understand the data at all. So, to convert that data’s in to the numerical values means there is a very famous method which is LabelEncoder.

This approach is simple and it involves converting each value in a column to a number. Taking a consideration of our dataset which has the categorical data. It will convert the value SE as to 1 and some other values to 2 and so on. If the same value appears again then it will take the same values and take the same output also like for SE it will take 1 only if it appears again in the dataset.

The dataset which have collected has many numerical values. In that the values are large like 100 or like this. If we give these values to our model then our model will be overfit. So, it will not generate the proper output bases on the given values. If we visualize the data the data will be varying from max to min values. To give it to the proper values and give the appropriate relation between the prediction values and the features. To take them in the relation we will use the Minmax scaler which is used to transform the values in to the between 0 and 1. When the values come to the values between 0 to 1 the model will give us the proper result.

Min Max scaling is the most used normalization technique in the Machine Learning. As we have discussed after scaling it will transform data to the range [0,1] meaning that the minimum and maximum values of feature/variable is going to be 0 and 1 respectively.

$$x_{scaled} = \frac{x - \min(x)}{\max(x) - \min(x)} \qquad \text{Eq} - \text{(i)}$$

Based on this equation (i) the values will convert between [0,1] and give it to the model for the creating a model.

As we have used the 2 famous algorithm of Recurrent Neural Network which are LSTM and GRU. We will discuss both the algorithm and understand the working of that both the algorithms.

The working of LSTM(Long Short term memory) network. The LSTM required because in the RNN we have problem of gradient vanishing. LSTM vary from more conventional feedforward neural networks in that they feature feedback connections. This LSTM characteristic allows it to analyse the full sequence independently, but it also uses knowledge of earlier data in the sequence to aid in processing new data points. LSTM model is reliant on these three factors :

- The current long-term memory of the network – know as the cell state
- The output at the previous point in time – known as the previous hidden state
- The input data at the current time step.

In total LSTM has three layers Input gate , forget gate and output gate.

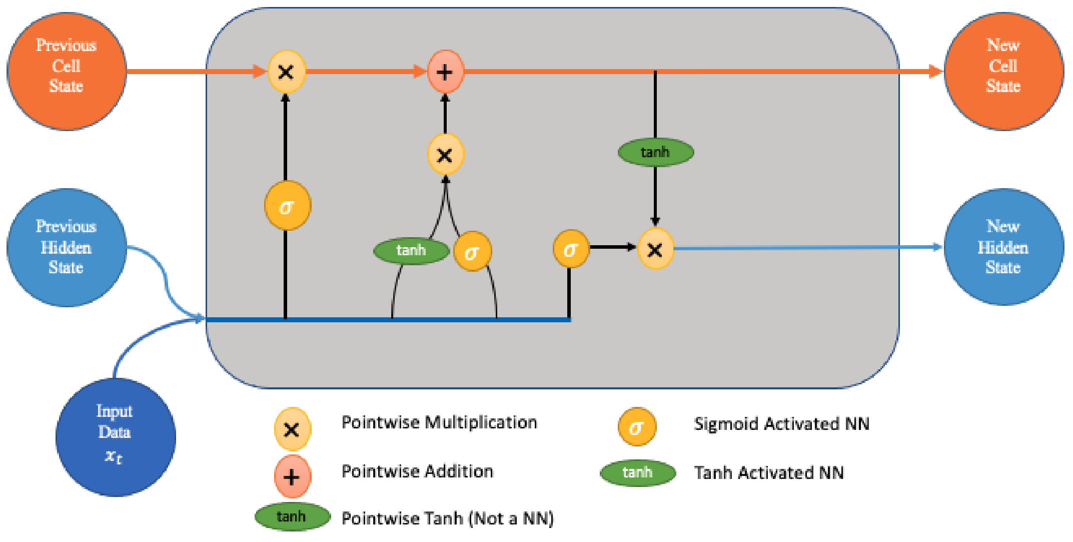


Figure 5.1.1 The LSTM architecture

As we can see in the figure that in LSTM we have 3 gate. LSTM using the sigmoid activation function for the in their gate architecture. If the outcome for the sigmoid is 0 then forget the data and if the outcome is 1 then put the data and go further step. In Forget gate, it will decide whether the data which we are providing for the processing are important or not based on the output of the sigmoid function.

In LSTM we have 3 gates namely forget gate, Input gate and output gate. Firstly, when data comes from the previous cell state it comes into the forget gate here, we are defining that weather we have to put the data or forgot the data. We will do that with the help of the sigmoid activation function (The output of sigmoid function will 0 or 1) after generating the output based on the input which we have received from the previous cell state. We will multiply that data to the current cell state So if the output will be 0 then we will forget the data or else if 1 then we will keep data and store into the current cell state. Here the task of forgot gate will over. Now the work for the input gate started and in that we will give the data from the previous cell state. Here firstly we will generate the output from the sigmoid activation function and along with that we will generate the output for the tanh (It will generate the output between -1 and +1) activation function. Then we will multiply it and add it to the current cell state. From the input gate we can understand that whether the data which we providing to the current cell state is important or not. Here the work of input gate will over. After that it comes the output gate in which we will first generate the output for the tanh activation function with the help of current cell state and then generate the output for the sigmoid of the previous cell state and then we will do product of that and send it to the next cell state. And then for the next cell state the process will continue like this.

When the data comes in to the cell state the first gate which the data will give it to is Forget gate. As the name implies that the gate will forget the data which comes in to their architecture. Here we will decide which bit of the cell state are useful give both the previous hidden state and new input data.

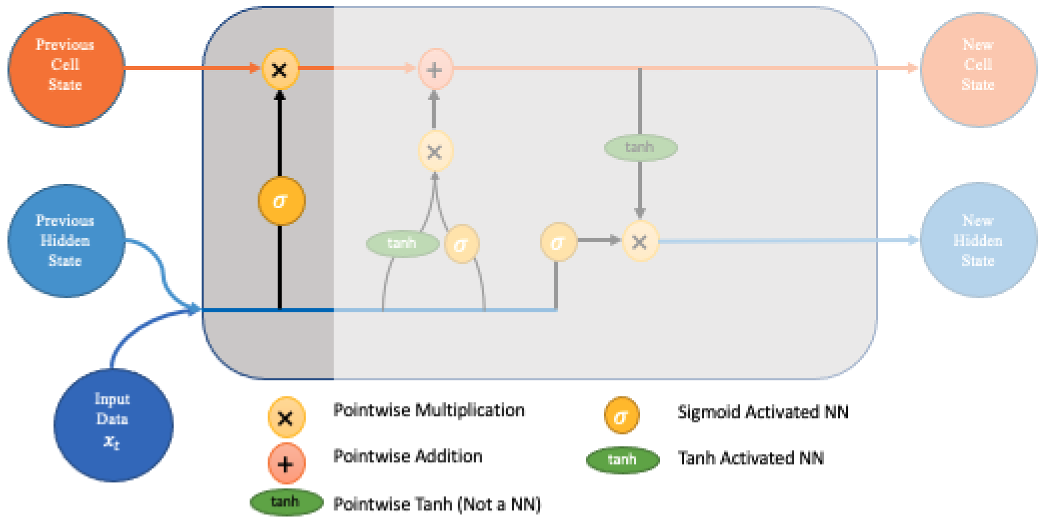


Figure 5.1.2 The Forget Gate for the LSTM

The neural network is fed both new input data and the previous hidden state. This network produces a vector in which each member is in the range [0,1]. (Guaranteed using Sigmoid activation). This network (in the forgetgate) is trained to output a value close to 0 if the input component is considered irrelevant and close to 1 if the input component is considered important. increase. It is useful to think of each component of this vector as a kind of filter or sieve that passes more data as the value approaches 1.

The sigmoid activation function converts the data between 0 and 1. That is helpful to update or forget data because any number multiplied by '0' will convert to 0, which can consider forgetting the data from the memory cell. And if the output is 1 then the value should be considered as the important data and kept that data in the memory cell. As we can see in the figure that the graph is from 0 to 1 and convert the values between 0 and 1.

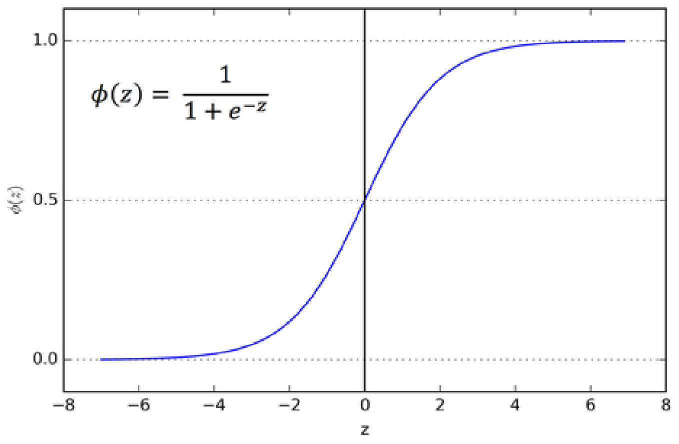


Figure 5.1.3 Showing the range of sigmoid activation function (0-1)

Following that, these output values are transferred upward and pointwise multiplied with the prior cell state. Because of this pointwise multiplication, elements of the cell state that the forget gate network has considered unnecessary will be multiplied by a value near to 0 and will therefore have less of an impact on the subsequent stages.

The second step us the input gate. The goal of this step is to determine what new information should be added to the networks long-term memory (cell state), given the previous hidden state and new input data.

At the first stage of input gate, the data which we will provide it to the current time stamp it will check whether the data what we have given it to the it will important to the model or not. If it is important then it will take that data and give it to the model for further training.

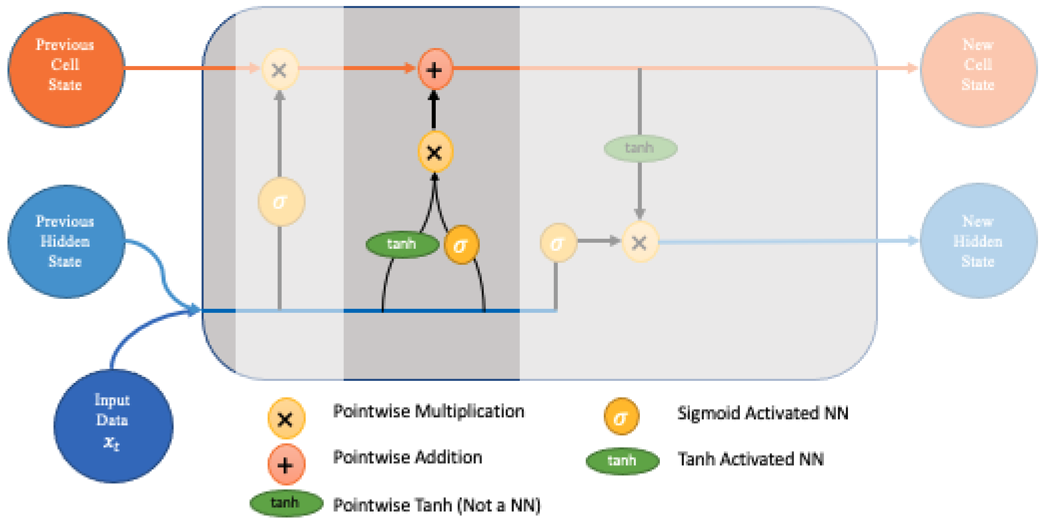


Figure 5.1.4 The Input gate of the LSTM

This can be done by the activation function called tanh activation function. It will create a new memory update vector by fusing the old concealed state with the fresh input data. Given the context from the prior concealed state, this vector effectively includes information from the new input data. Given the new information, this vector indicates how much to update each component of the network's long-term memory (cell state)..

Notably, we employ a tanh since its values fall between [-1,1] and can, thus, be negative. If we want to lessen the influence of a component on the cell state, the possibility of negative values is required here.

As similar to the sigmoid Activation function even tanh is also playing a major role in the LSTM memory cell. Tanh will squish the values between -1 and 1 and this is useful in the Input gate and convert the values between -1 and 1 and multiply with the existing data of that Input gate.

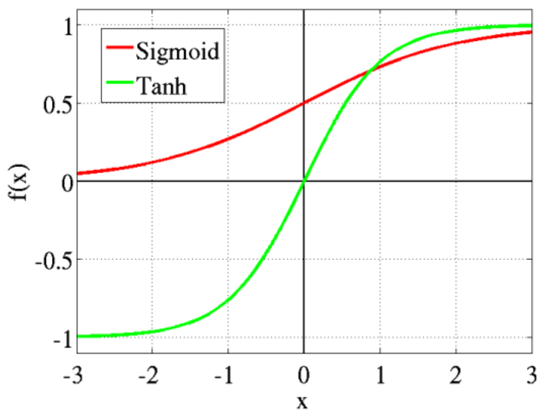


Figure 5.1.5 Comparing the range of activation function and sigmoid activation function

At the second stage of input gate, it will check for the data whether it is important or not, at the first stage it will not check for that. In this second stage the sigmoid (0,1) activation functions come in to the picture from this we can understand the data what we have provided are valid data or not or whether we have to put that data for the model or leave it.

The output for this input gate will be product of the first stage output and second stage output. The resulting combine vector is then added to the cell state, resulting in the long-term memory of the networking in the long-term memory of the network being updated.

At the final we have output gate which will decide the new hidden state. Three factors—the recently updated cell state, the prior hidden state, and the new input data—will be used to decide this. For this we will apply filter same as the forget gate. The inputs are the same and the activation function is also sigmoid ([0,1]).

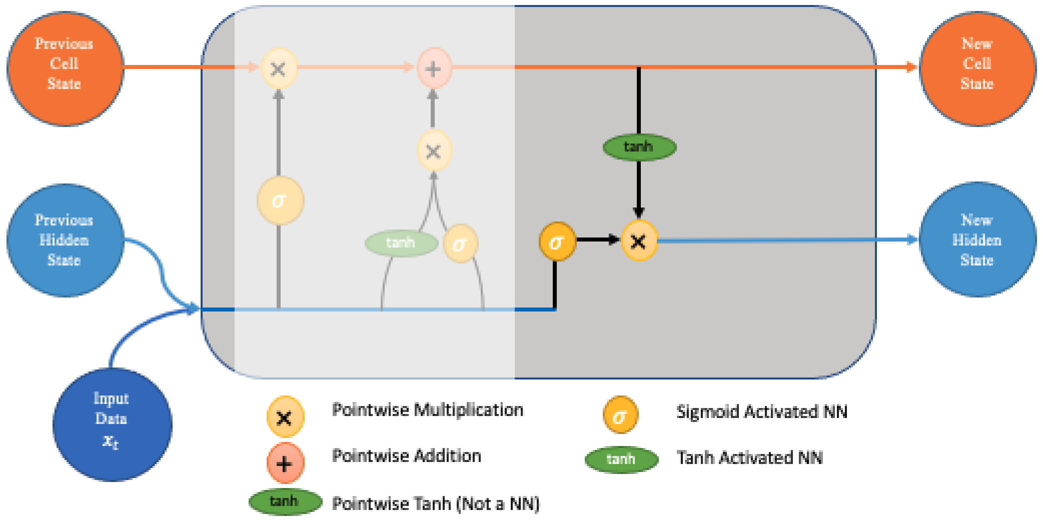


Figure 5.1.6 Output Gate for the LSTM architecture

The output gate will apply the filter to the newly updated cell state. This will take care that only necessary information will go in the output. Before applying the filter , we pass the cell state through the tanh activation function to force the values to change into the intervals of -1 and 1.

Same for the GRU (Gated recurrent unit). We can consider this as same as the LSTM because both are build on the same architecture. The GRU comes into the picture because in LSTM it is taking too much time and more computational power also to generate model and calculate all the values and create a pattern.

In GRU we only have 2 gates which are reset gate and update gate. The reset gate is similar to the forgot gate in the LSTM, it will justify whether the data from the previous cell state we have to keep it or reset that cell state. After that it comes the update gate. In update gate we will update the cell state with the help of sigmoid activation function and add the output of that with -1 and multiply it to the cell state. For the output of the Memory cell it will multiply with tanh activation function and then multiply with the update gate output and add it to the current cell state.

The architecture for the GRU are look like this.

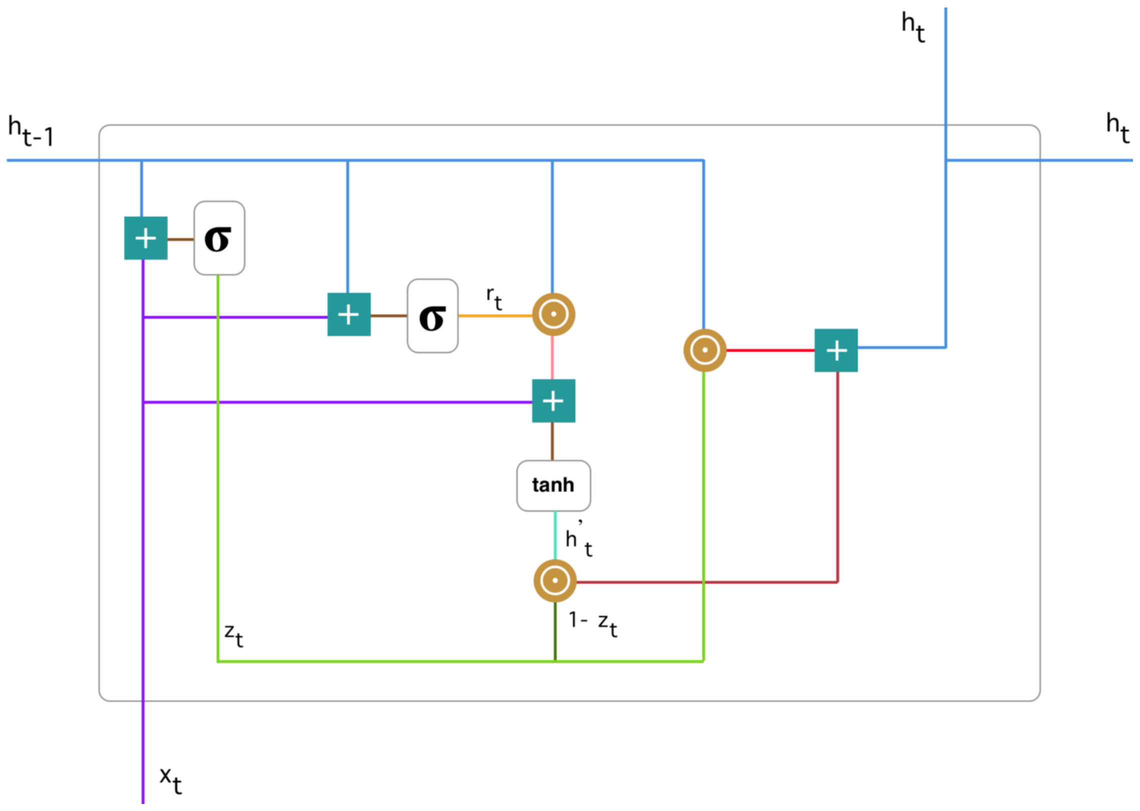


Figure 5.1.7 The GRU architecture

The GRU has manly only 2 gates which are reset gate and update gate. In this we are taking help of both sigmoid and tanh activation function which will help us to understand the data and making the decision for whether we have put the data or we have reset the data in the architecture.

The first step will be update gate It will aid the model in determining how much information from the past needs to be transmitted to the future. That is particularly potent since the model has the option of copying all historical data, which avoids the danger of the vanishing gradient problem. The formula for calculating the update gate is :

$$z_t = \sigma(w^{(z)}x_t + U^{(z)}h_{t-1}) \qquad \text{Eq – (ii)}$$

When x t is connected to the network unit, its weight W is doubled (z). The same is true for h (t-1) which is multiplied by its own weight U and contains information for the preceding t-1 units (z). Together, the two results are summed, and the result is then squeezed between 0 and 1 using a sigmoid activation function. The following schema is what results:

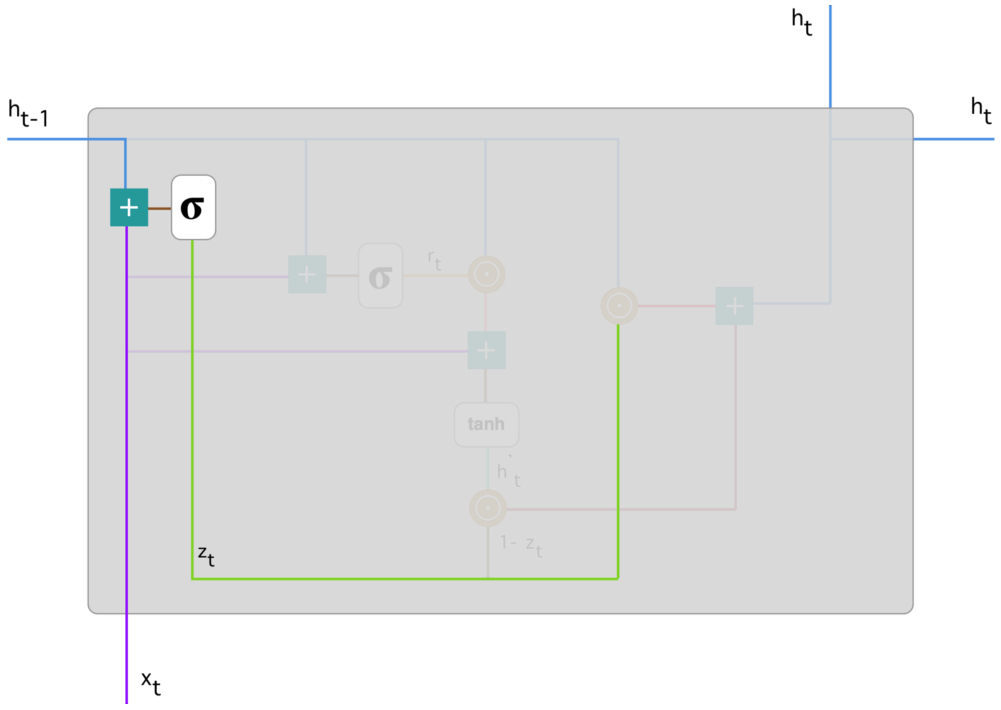


Figure 5.1.8 Update gate for the GRU

The second gate will be of reset gate, this gate is used to determine how much the model forgets past information. It means it will determine which data is important in the form of the model creation and which information if we forget that it will not affect the model. The equation for the reset gate will be :

$$r_t = \sigma(W^{(r)}x_t + U^{(r)}h_{t-1}) \qquad \text{Eq – (iii)}$$

This formula is the same as the one for the update gate. The difference comes in the weights and the gate's usage, which will see in a bit. The schema below shows where the reset gate is:

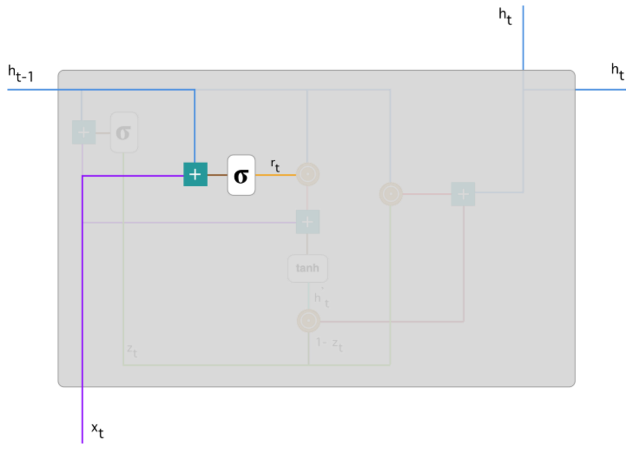


Figure 5.1.9 Forget gate for the GRU

Now that we know precisely how the gates effect the model's output, we can make more informed decisions. We completed the reset gate in the first phase. We present new memory content that will utilise the reset gate to retain the necessary historical data. It is calculable as :

$$h'_t = \tanh\,(Wx_t + r_t * Uh_{t-1}) \qquad \text{Eq} - (\text{iv})$$

The second is an update gate vector that holds the information for the current entity and forwards it to the network. This requires an update gate. Decide what to collect from the current memory contents (h`_t) and what to collect from the previous step (h_ (t-1)).That is done as follows:

$$h_t = z_t * \ h_{t-1} + (1 - z_t) * h'_t \qquad \text{Eq} - (\text{v})$$

We have used both the LSTM and GRU models for forecasting data. In our model, you have to give the last 15 days of data with the parameters of {date, pollution, dew, wind_dir, wind_spd, snow, rain, pollution} based on that we can forecast the next 24 hours of data which is pollution. We have to build a model to predict the next 24 hours of data. In this, the date is playing the most important role in forecasting data.

5.2 IMPLEMENTATION

For implementing the LSTM and GRU algorithm we have used TensorFlow library. For implementing this we have followed several steps which are as follow:

a) Data Pre-processing
b) Creating Model
c) Saving Model
d) Generate Output

a) Data Pre-processing:

For Data Pre-processing firstly we have to understand the data which we have. To understand the data we can use the EDA. Firstly if in our data set there is a requirement for the Encoding of data we have to do that we can do that with the help of a Label Encoder or One Hot Encoder. After that, we have to normalize the data so our model can learn from that without getting more confused.

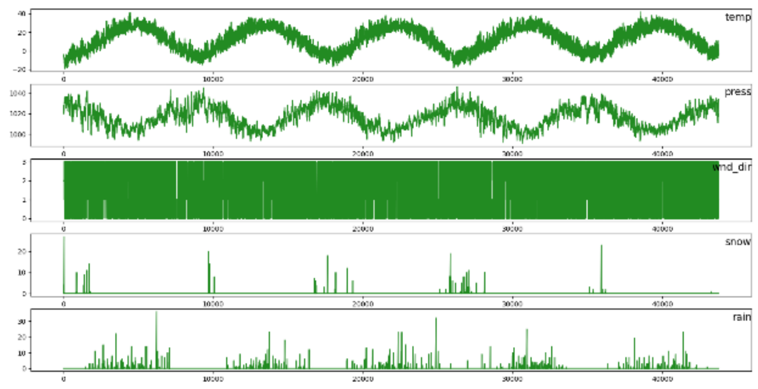


Figure 5.2.1 Represent of data variable from dataset

This graph in figure 13 represents the input variable data from the dataset and visualize it. It will easy to understand the data from which we can organize the data.

For Label Encoding process of the data the data should be processed and generate the one unique number for the each and every unique data in the column of that dataset. But in the case of One Hot Encoder the for every unique data in the column It will generate one number and rest of them assign to the same values.

The label encoder is required the dataset column name wnd_dir as in that we have categorical values for the wind directions. We can achieve this by implementing using the sklearn library and in that library we have the method with the name of Label Encoder. Before implementing this to our dataset in total we have 4 types of data in the wind direction column with name :

(i) SE (South East wind)
(ii) CV (Calm and Variable wind)
(iii) NW (North west wind)
(iv) NE (North East wind)

So if we will give this values to our model it won't accept the data itself. Because our model need the mathematical data to understand the relation between the feature values and prediction values. For that we need to encode the data with the help of Label Encoder.

For that we will load the Label Encoder from the sklearn firstly after than we will convert the text values into the numerical values.

```
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
dataset[dataset_index] = encoder.fit_transform(dataset[dataset_index])
```

After implementing Label Encoder to our dataset column our dataset will look something like this:

(i) SE will convert to 2
(ii) CV will convert to 3
(iii) NW will convert to 1
(iv) NE will convert to 0

Now in our dataset we have only numerical values in the entire dataset we have converted that one text data into the numerical values. But when it comes to the dataset we can see that we have the values which are too big for our model. The data which we have has many variations in it. Because of that our model cannot learn the relation between the prediction values and features values. To overcome that we will normalize the data into the smaller values.

To do that we have used the Min Max scaler that is used to scale the data in to the variation of 0 and 1. Because of the scaling the data the all the data will come into the range of 0 and 1 so our model can understand the relation between the data in the dataset.

To implement this firstly we have to load the dataset in the induvial variable and transform it to the range of 0 to 1. But before that we have to do the reshaping of the data with the help of reshape method. We have to do this for the all the features available in the dataset.

As in our dataset we have in total 11 features and 1 prediction variable So we have to do the normalization of all the 11 features for our dataset.

```
x_1 = dataset['dew'].values
x_2 = dataset['temp'].values
x_3 = dataset['press'].values
x_4 = dataset['wnd_spd'].values
x_5 = dataset['wnd_dir'].values
x_6 = dataset['snow'].values
x_7 = dataset['rain'].values
x_8 = dataset['year'].values
x_9 = dataset['month'].values
x_10 = dataset['day'].values
x_11 = dataset['hour'].values
y = dataset['pollution'].values #Prediction value
```

As we can see we have loaded all the dataset feature and prediction values to the specific variables.

If we visualize our dataset based on the actual values it has many large numbers which can lead to the problem for us. The data we can see as below :

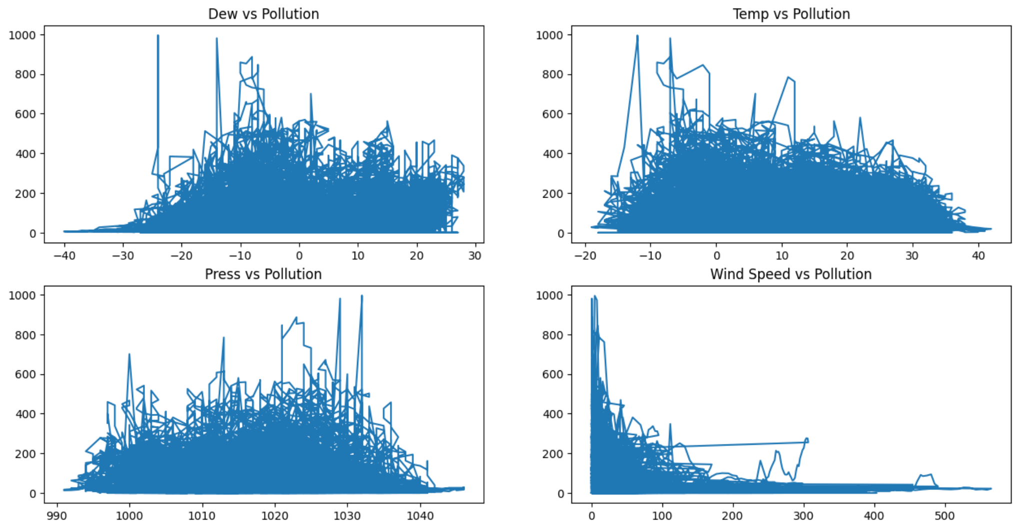


Figure 5.2.2 Dataset before normalizing data

As we can see that all the values are ranging from the 0 to 1000 for the pollution in the y-axis data and for the x-axis we have different features which are dew, temp, press and wind speed. We have only take the 4 parameters for this visualizing the data.

This below graph in figure 15 represent the data for the output variable for the last 15 days of data with respect to every hour which mean it shows the data of 360 hours. But in this we have plot without scaling the data. As we can see in the y-axis the value of the air pollution with respect to the number of hours which is 360.

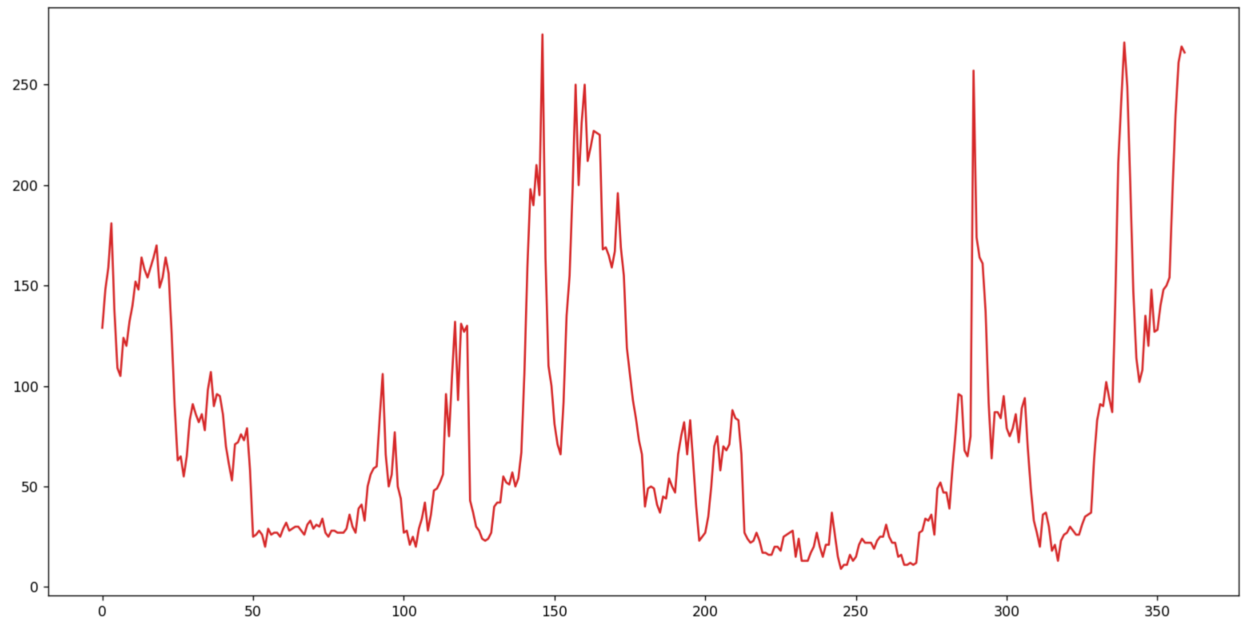


Figure 5.2.3 Data visualization for the Air Pollution on the hourly basis for the last 15 days which is 360 hours of data

For selecting the features which are appropriate for the model we have to understand the co-relation between the data. From that we can select the features which are useful to us. If we will select the features more which are highly co-related to the each-other then our model will come in the state of overfitting. Which means our data will work more accurate with the training data but when it comes to the testing data it will not predict accurate.

We have created the co-relation metrics for the dataset and but the dataset is not in the normalized data. From this we can understand the dataset in the manner of how well the dataset are interconnected with each other. It will compare all the dataset column with each other and generate the graph with the values between 0 and 1. 0 means the features are not well co-related and the values 1 means the features are highly co-related.

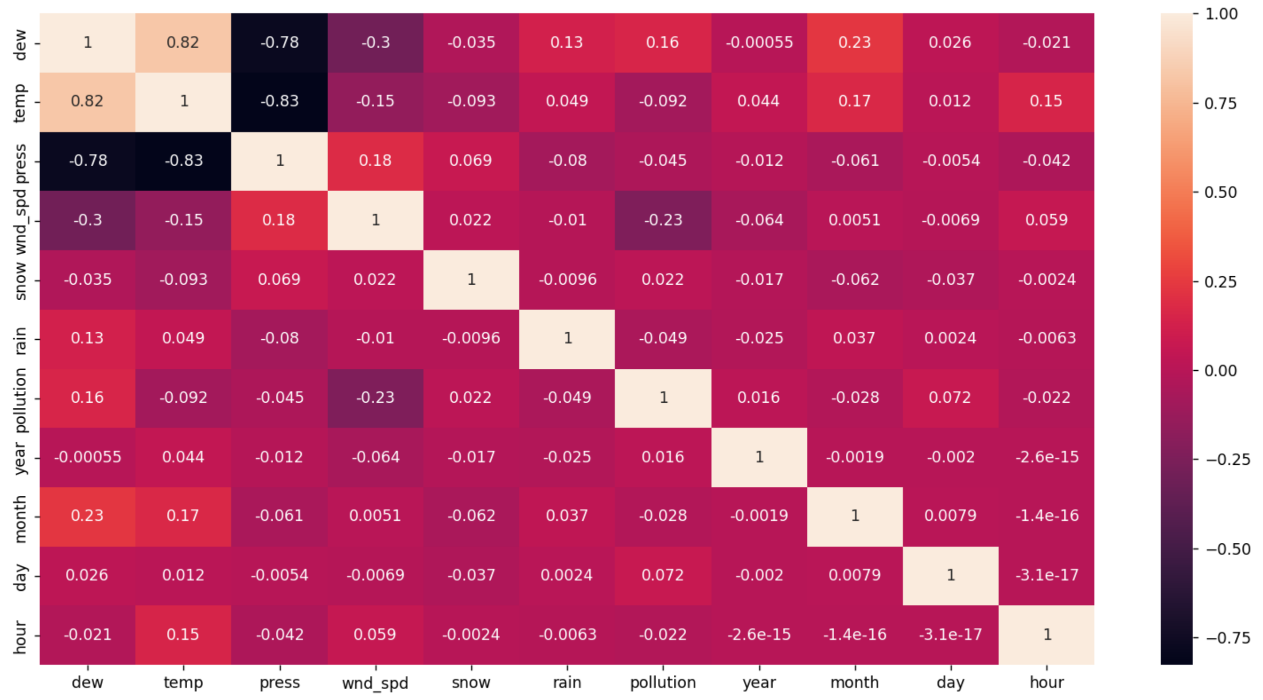


Figure 5.2.4 Heat map for the co-relation of data of the dataset

Now we will do the normalization of data with the help of Min max scaler of all the features and the output variable also. We can do that by applying this code.

```
scaler = MinMaxScaler(feature_range=(0, 1))
x_1_scaled = scaler.fit_transform(x_1)
x_2_scaled = scaler.fit_transform(x_2)
x_3_scaled = scaler.fit_transform(x_3)
x_4_scaled = scaler.fit_transform(x_4)
x_5_scaled = scaler.fit_transform(x_5)
x_6_scaled = scaler.fit_transform(x_6)
x_7_scaled = scaler.fit_transform(x_7)
x_8_scaled = scaler.fit_transform(x_8)
x_9_scaled = scaler.fit_transform(x_9)
x_10_scaled = scaler.fit_transform(x_10)
x_11_scaled = scaler.fit_transform(x_11)
y_scaled = scaler.fit_transform(y)
```

As we can see that the feature range is varying from the 0 to 1. After that we will apply fit_transform for all the variables. After this all the data are converted into the range of 0 and 1 and we will store this all the variable into the new dataset with the help of hstack. And we will create another data frame for our dataset.

After that if we visualize the data which are normalized will look something like this.

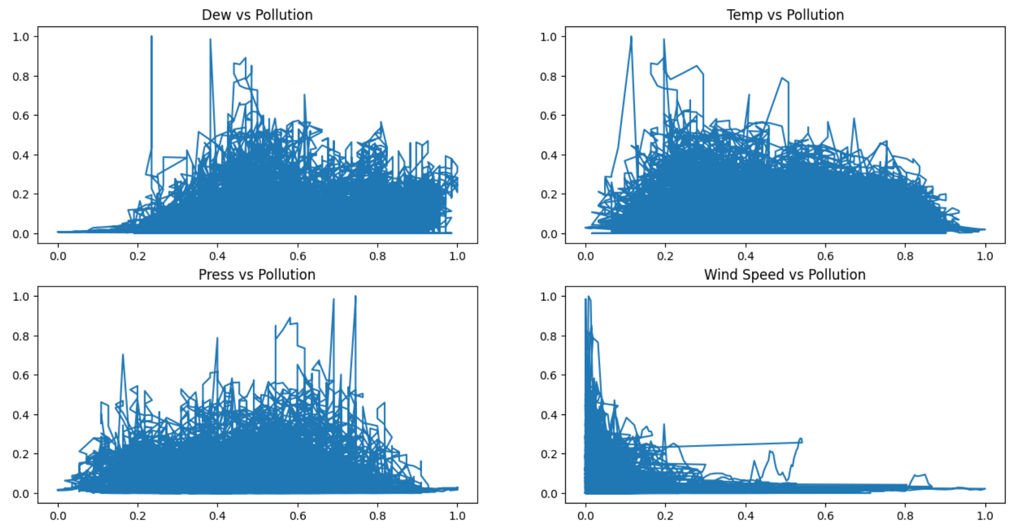


Figure 5.2.5 The dataset visualization after normalizing the data

Here we can see the data are ranging from the 0 to 1 for both the axis x and y. Here we can see that for the all the four input parameters and output parameters. Now we can see that all the ranges are from 0 to 1. And for the same for the prediction data variable it will generate the graph for the prediction value with the normalized data from the dataset.

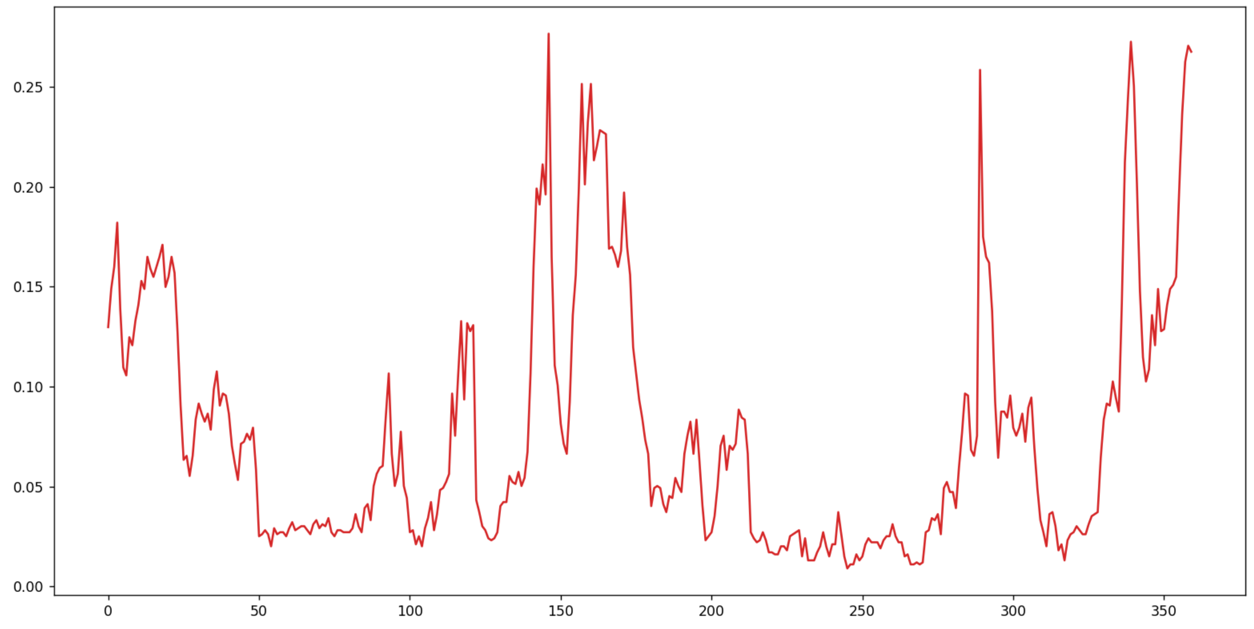


Figure 5.2.6 The Normalized data for the prediction variable pollution

As we can see that now the values are range from the 0.0 to 0.30 values. From the 15 days of data which is 360 hours of data on the hourly basis. Now if we will give this data to the model or model can learn more relation between the dataset and can generate the more efficient model.

b) Creating a model:

In total we have created 2 models which are LSTM model , GRU model.

For creating the GRU model , we have to create a GRU model. For this also we have used Input layer of 360 neurons which is 15 days on hourly basis. Two hidden layers with each of 50 neurons in it. And also added the Dropout layer which is also required to add.

The model architecture would look like this for the GRU model.

Layer (Type)	Output shape	Param #
gru (GRU)	(None , 360 , 50)	9450
dropout (Dropout)	(None , 360 , 50)	0
gru_1 (GRU)	(None , 360 , 50)	15300
dropout_1 (Dropout)	(None , 360, 50)	0
gru_2 (GRU)	(None , 50)	15300
Dense (Dense)	(None , 24)	1224
Activation (Activation)	(None , 24)	0

Table-5.2.1 Model Architecture for GRU

In this architecture we have 2 layers of GRU and 2 layers of Dropout layer with 0.2. The input layer has the input shape of the (360,11). With the activation function of tanh and return activation function of sigmoid. We have added the dropout layer because we will remove some neurons which are not useful and the neurons which are not connecting in the next layer that kind of neurons are not important for us. So we will drop that neurons to reduce the tense of model. After that it start with the hidden layers of model in that the first hidden layer has the data with the 50 neurons in it. It will take the data as input from the previous layer. For the output layer we have used the dense layer with the activation layer of the linear activation function. Same as for the second hidden layer. For this architecture the total parameters we have is 41,274 and from that trainable parameter are 41,274. In this architecture also we have to provide the input for the last 15 days of data on the hourly basis and it will forecast the next 24 hours of data according to GRU model architecture.

For this to compile the model we have used the Mean square Error for visualizing the error rate for the model. And the optimizer of the adam optimizer with the learning rate for 0.001. And the metrics of the accuracy.

```
opt = keras.optimizers.Adam(learning_rate=0.001)
# define model
model = Sequential()
# Creatin a model
model.add(GRU(50, activation='tanh' , recurrent_activation='sigmoid' ,
return_sequences=True , input_shape=(n_steps_in, n_features)))
model.add(Dropout(0.2))
model.add(GRU(50 , activation='tanh' , recurrent_activation='sigmoid' ,
return_sequences=True ))
model.add(Dropout(0.2))
model.add(GRU(50 , activation='tanh' , recurrent_activation='sigmoid'))
model.add(Dense(n_steps_out))
model.add(Activation('linear'))
model.compile(loss='mae', optimizer=opt , metrics=['accuracy'])
```

For creating LSTM model, we have used the same parameters like GRU model creation. For creating a LSTM model, we have used the same layers for like Input layer, hidden layers and output layer. Firstly, the input layer is of 360 neurons and alongside of that we have 2 hidden layers each of 50 neurons and 2 layers of Dropout layer at the end we have Dense layer which is output layer.

The architecture for the LSTM model look like this

Layer (Type)	Output shape	Param #
lstm (LSTM)	(None , 360 , 50)	9450
dropout (Dropout)	(None , 360 , 50)	0
lstm_1 (LSTM)	(None , 360 , 50)	15300
dropout_1 (Dropout)	(None , 360, 50)	0
lstm_2 (LSTM)	(None , 50)	15300
Dense (Dense)	(None , 24)	1224
Activation (Activation)	(None , 24)	0

Table-5.2.2 Model Architecture for LSTM

For this to compile the model we have used the Mean square Error for visualizing the error rate for the model. And the optimizer of the adam optimizer with the learning rate for 0.001. And the metrics of the accuracy.

```
opt = keras.optimizers.Adam(learning_rate=0.001)
# define model
model = Sequential()
```

```
# Creating a model
model.add(LSTM(50, activation='tanh' , recurrent_activation='sigmoid' ,
return_sequences=True , input_shape=(n_steps_in, n_features)))
model.add(Dropout(0.2))
model.add(LSTM(50 , activation='tanh' , recurrent_activation='sigmoid' ,
return_sequences=True ))
model.add(Dropout(0.2))
model.add(LSTM(50 , activation='tanh' , recurrent_activation='sigmoid'))
model.add(Dense(n_steps_out))
model.add(Activation('linear'))
model.compile(loss='mse' , optimizer=opt , metrics=['accuracy'])
```

c) Saving model:

The creation of model is time consuming process. To avoid that process to do it every time we will save that model for the future use. In future if we have to use that model we can use that easily by just calling that model to our application. By the help of saving model we can use that model in any kind of application like web application , we can create a API from that and many more use case.

For creating a model we have to fir the model based on the given parameters like epoch values and step_per_epoch values and other values.

```
history = model.fit(train_X , train_y , epochs=300, steps_per_epoch=25 ,
verbose=1 ,validation_data=(test_X, test_y) ,shuffle=False)
model.save('Air_Pollution.h5')
```

We can save the model with the help of this code. And the name for the model will be the Air_pollution.h5, and the model has saved for the future use.

d) Generate output:

The output which will generate from this model are next 24 hours of air pollution data. We just have to provide the input of the last 15 days of data with respect to the every hour which is in total 360 hours of data. We have to give last 360 hours of data and it will predict the next 24 hours of data.

To generate the output we have to first load the model which we can do by the help of tensorflow.

```
model = load_model("Air_Pollution.h5")
```

By this we can load the model which we have saved in the local machine. After we have to follow some steps for the data pre-processing which will pre-process the data which we have to give for the forecasting same as that we have to do scaling of data and even that we have to do normalization of data. After that we have to convert that data into the 1 Dimensional numpy array. Then we can give that model for the forecasting and based on that it will forecast the data.

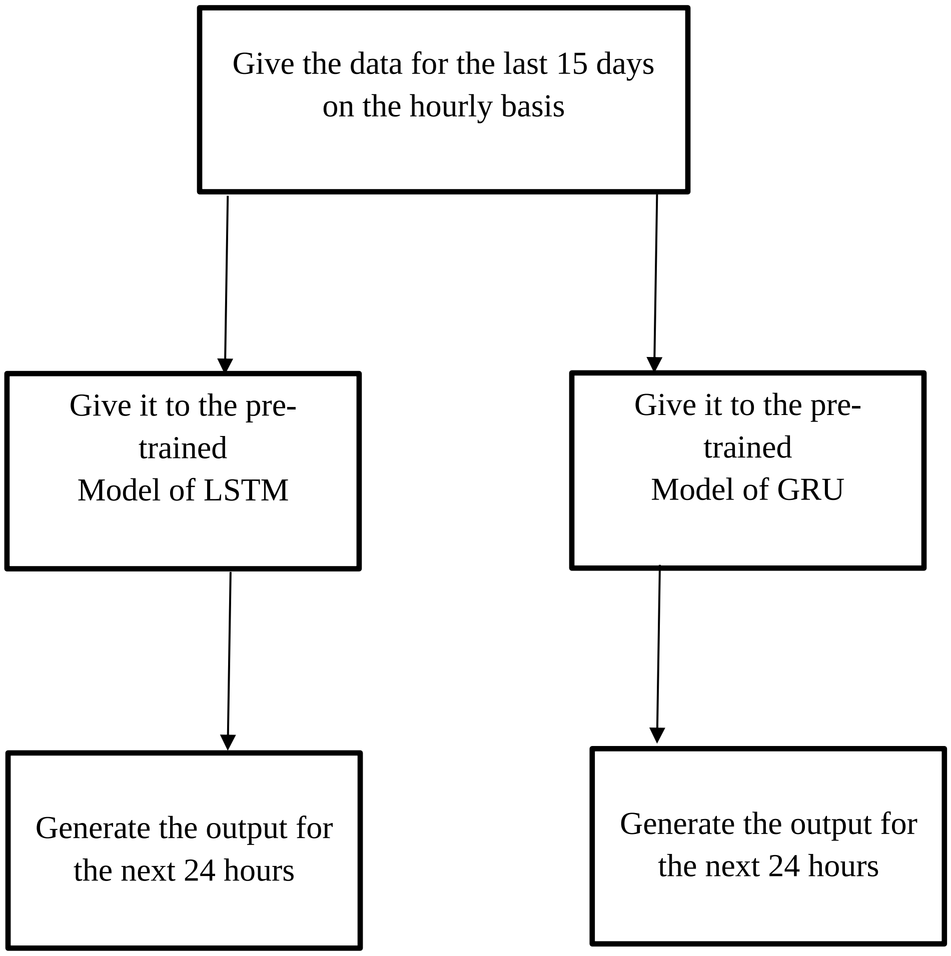
```
y_pred = model.predict(data)
```

Based on this it will predict the next 24 hours of data. But the data will be in the range of 0 to 1. Because we have normalized the data but we want the actual output of data. For that we have inverse transform the data with Min Max scaler as what they have predicted. We can achieve that by doing this :

```
y_pred_inv = scaler.inverse_transform(y_pred)
```

6. SYSTEM DESIGN

6.1 HIGH LEVEL DESIGN


Give the data for the last 15 days on the hourly basis
Give it to the pre-trained Model of LSTM
Give it to the pre-trained Model of GRU
Generate the output for the next 24 hours
Generate the output for the next 24 hours

7. RESULT

We have trained the model on the basis of 3 types of epoch number 100, 200 and 300. For every model we have done this and based on this we have selected the best model which will more accurate for us. The all the 3 models has generated the output graph which will saw us how accurate is our model in every epoch levels.

Firstly we will go with the GRU model we have trained that model with the 3 epoch values which are 100, 200 and 300. With the 50 neurons in every layer. And based on that it has generated the output values and graphs. The first graph will represent that epoch values as we have train the model for the 300 epoch firstly we have said that how well the model has fit for the epoch values.

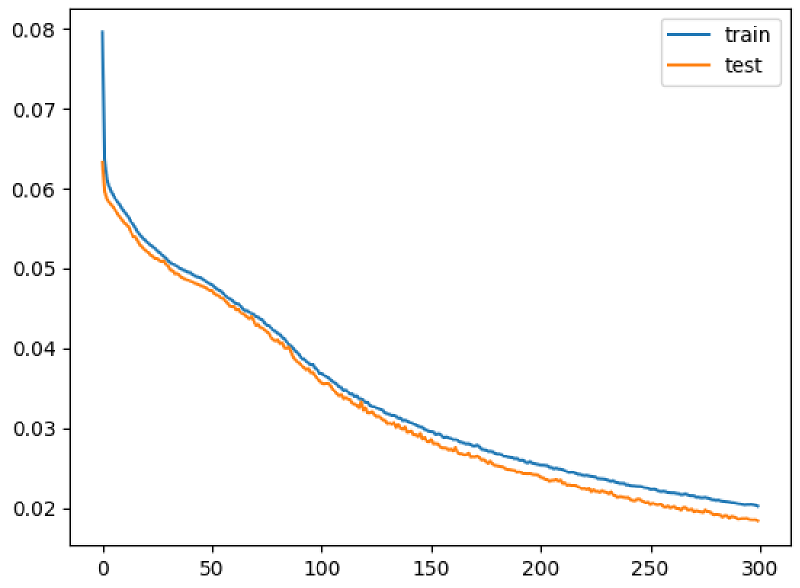


Figure 7.1 The graph for the 300 epoch values for the GRU with 50 neurons

And based on that trained model we have to predict the values by providing the values for the last 15 days of the data for the hourly basis. Which means in total we have 360 hours of data which we have to provide for the forecasting. The GRU based model has generated the accuracy score for the 300 epoch is 91.0987%.

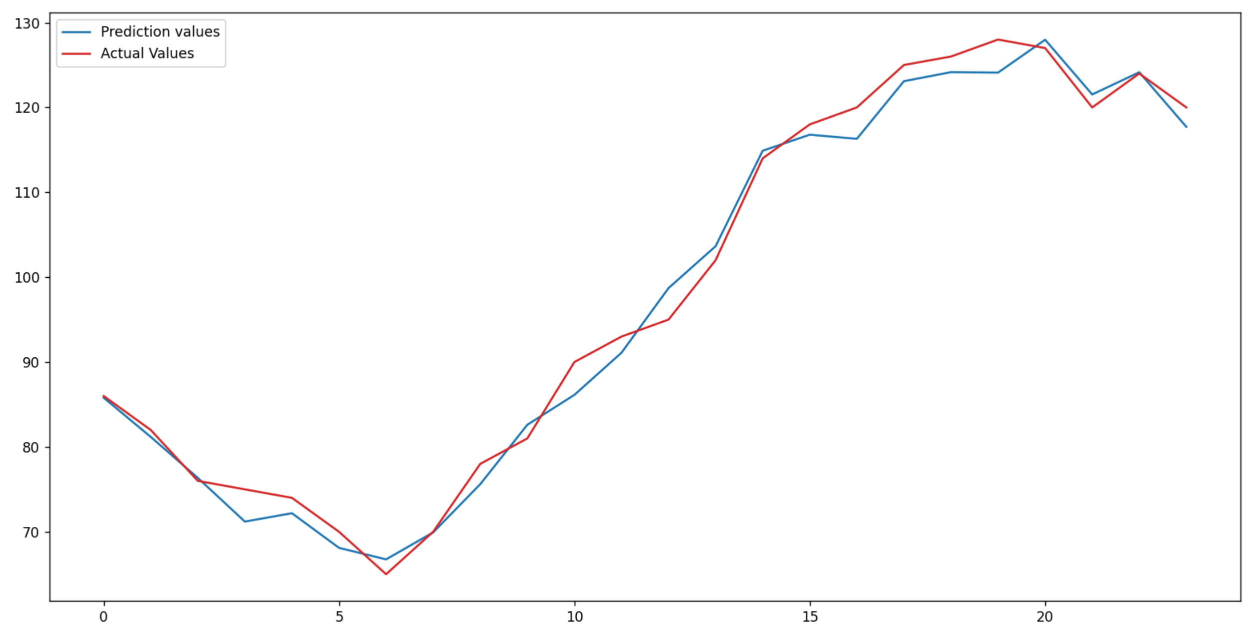


Figure 7.2 The forecasting of data for the next 24 hours data based on the data what we have provided for 50 neurons.

Same like this we have trained the model for the GRU and with the epoch values of 100 and 200 and generated the output for the same which is next 24 hours of data.

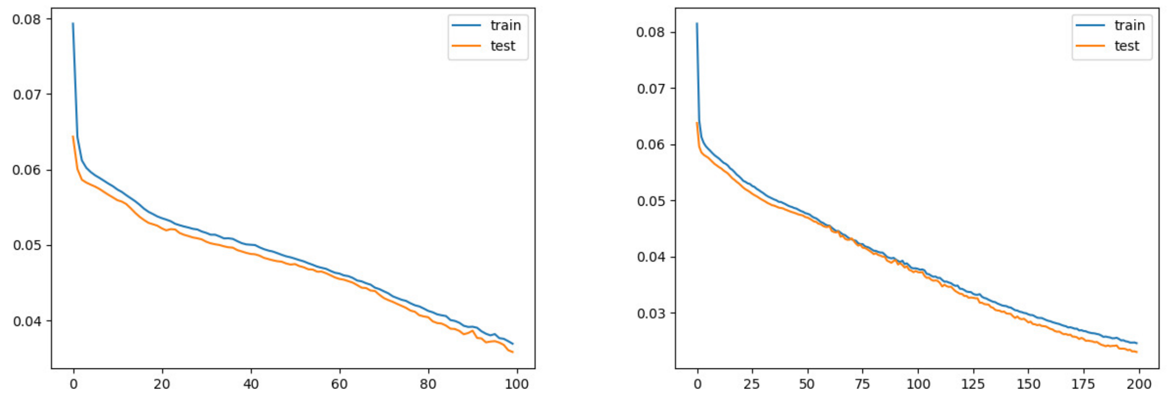


Figure 7.3 The graph of 100 and 200 epoch values for the 50 neurons for the GRU

The model what we have saved for both the 100 and 200 epoch values and based on that generated the output for the air pollution data. We will give the same data for the forecasting and it will generate the output. The output will generate in the graph and we have saved that model. The 100 and 200 epoch values accuracy score are 44.0876% and 68.956%.

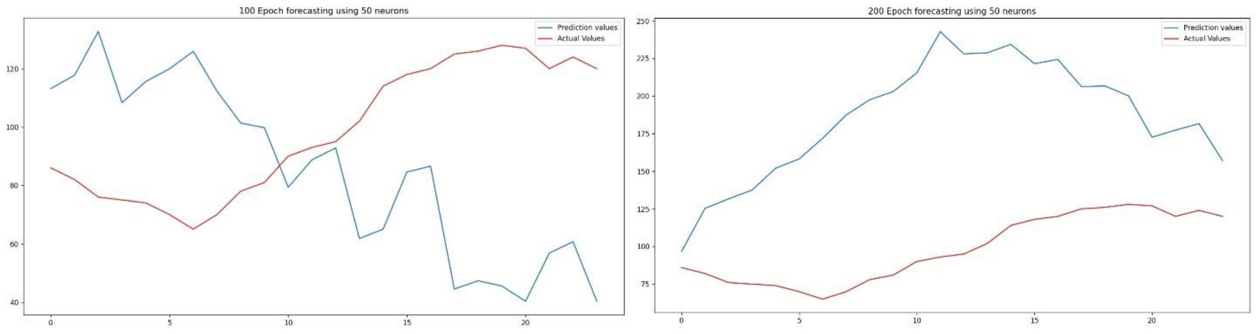


Figure 7.4 The graph for the both the model with 100 and 200 epoch values.

From the above given data we can see that model with 300 epoch values is performing well from all the 3 models of 100, 200 and 300 epoch values.

For the proposed system the model for the GRU is ready and now for that we will create LSTM model with the same configuration of the model just we have swapped the GRU layers with the LSTM layer and also for that the same configuration for the LSTM also with 100, 200 and 300 epoch values. For this LSTM model also we will train all the 3 model values. This is the first graph for the 300 epoch values for the LSTM model. The LSTM based model accuracy score Is 92.876%.

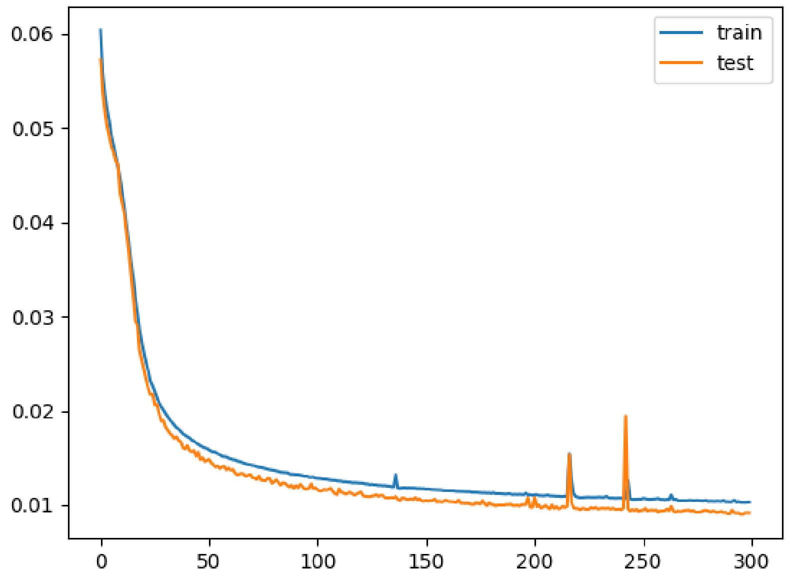


Figure 7.5 The train and test validation data for the model with 70 neurons and 300 epoch

The output what we have generated using the LSTM model and 300 epoch values for that the graph is in Figure 24. This is the same as we have to give the last 15 days of data for the every hour and it will generate the next 24 hours of data. It will generate the graph for the prediction value and Actual value. The graph for this model will look like this.

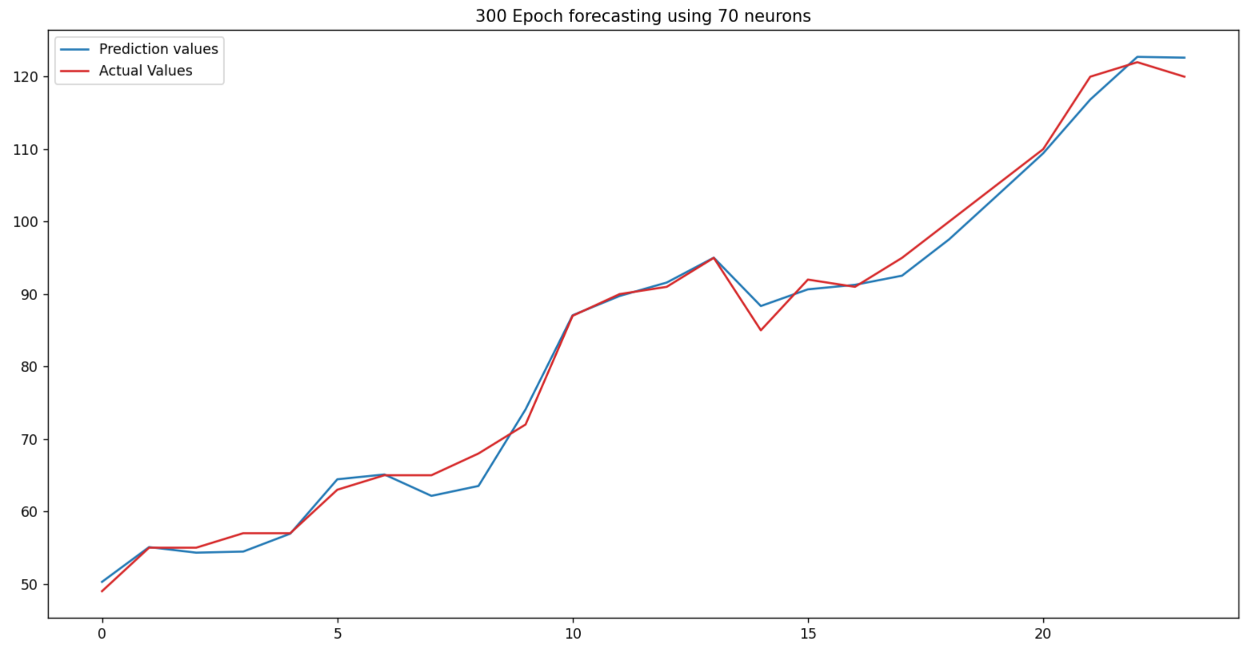


Figure 7.6 The output graph for the next 24 hours of data for the 70 neurons

Same for the GRUalso we have trained the 3 model in total with the 100, 200 and 300 epoch values from that 300 epoch values model output you can see in the figure 24 and now we will go with the model of 100 and 200 epoch values. The model architecture is same as earlier just we have done is that train the model with 100 and 200 epoch. The model training graph for both the epoch values you can see here

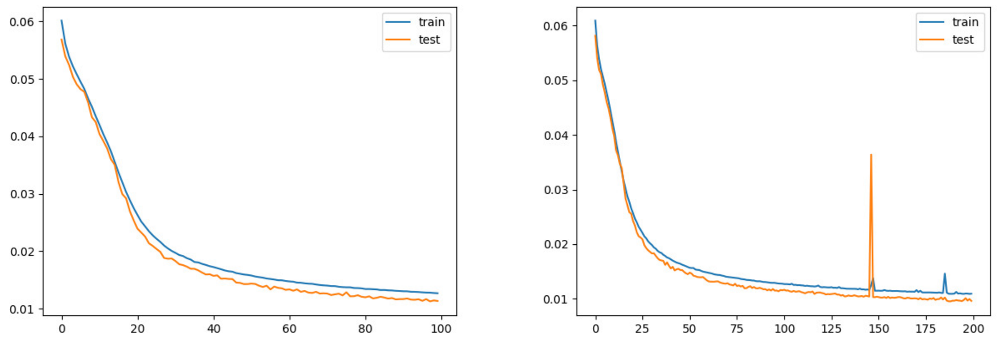


Figure 7.7 The training graph for the 100 and 200 epoch with 70 neurons

Based on the trained model the graph has generate and model have saved. But to see how accurate our model is we have done the prediction for the air pollution and created the graphs for that the graph is for both the prediction and accurate values. That graph for both the model predictions and actual values are. The 100 and 200 epoch values accuracy score are 49.7098% and 72.476%.

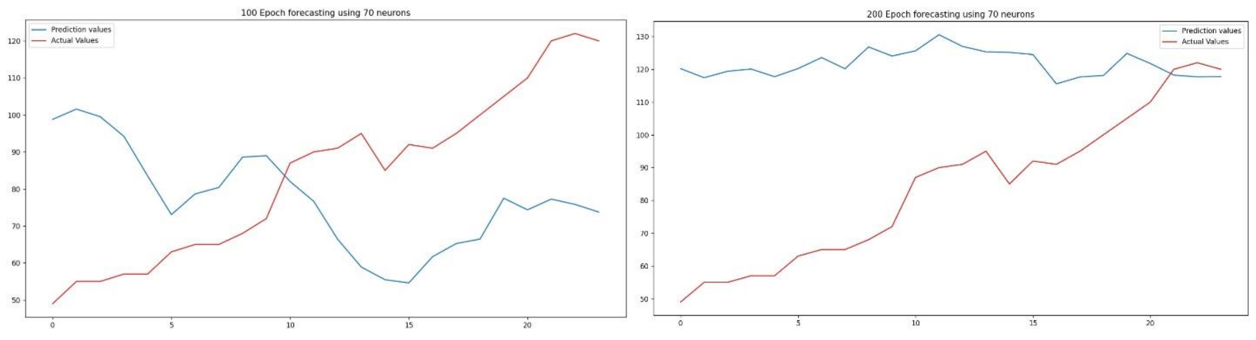


Figure 7.8 The output prediction graph for the actual and predicted values

8. CONCLUSION AND SCOPE FOR FUTURE ENHANCEMENT

With the help of this models, we can forecast the air pollution data for the next 24 hours by providing the last 15 days of data to it. As we have created 3 models for the forecasting of data that are LSTM, GRU. This will generate the output for the next 24 hours of data and the data are quite accurate also. From this everyone can see the forecasting of air pollution along side the weather forecasting which will help the users to understand the air pollution out there so he/she can take precaution so he/she can be safe from the diseases which are caused due to inhaling polluted air.

The dataset collected is of only 4 years if we can collect more years of data then our model can be more accurate while forecasting the values. For that we have to continuously collect the data from the real time and we have to store for the daily basis. And here we have used the old dataset for the 4 years which is 2010 to 2014. But now the scenarios are different if we are able to collect this time of data and give it to the model than our model can work in the real time. And for the second thing we can give the data to the model and we will do on the daily basis then our model can even learn more about the patterns.

In Future we will feed more data to the model from which our model will become more accurate and learn more patterns from the data and our model will become more accurate. And also we are planning to launch this application as API from which anyone can access this API and use it to their application to show the data.

9. BIBILOGRAPHY

[1] K Srinivasa Rao, G. Lavanya Devi, N. Ramesh, "Air Quality Prediction in Visakhapatnam with LSTM based Recurrent Neural Networks", International Journal of Intelligent Systems and Applications(IJISA), Vol.11, No.2, pp.18-24, 2019. DOI: 10.5815/ijisa.2019.02.03.
https://www.mecs-press.org/ijisa/ijisa-v11-n2/IJISA-V11-N2-3.pdf

[2] Belavadi, Sagar & Rajagopal, Sreenidhi & R, Ranjani & Mohan, Rajasekar. (2020). Air Quality Forecasting using LSTM RNN and Wireless Sensor Networks. Procedia Computer Science. 170. 241-248. 10.1016/j.procs.2020.03.036
https://www.researchgate.net/publication/340636552_Air_Quality_Forecasting_using_LSTM_RNN_and_Wireless_Sensor_Networks

[3] Q. Tao, F. Liu, Y. Li and D. Sidorov, "Air Pollution Forecasting Using a Deep Learning Model Based on 1D Convnets and Bidirectional GRU," in IEEE Access, vol. 7, pp. 76690-76698, 2019, doi: 10.1109/ACCESS.2019.2921578.
https://ieeexplore.ieee.org/document/8732985

[4] X. Du, Y. Cai, S. Wang and L. Zhang, "Overview of deep learning," 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), 2016, pp. 159-164, doi: 10.1109/YAC.2016.7804882.
https://www.mdpi.com/2227-7390/10/13/2245/pdf?version=1656327047

[5] J.B. Ordieres, E.P. Vergara, R.S. Capuz, R.E. Salazar,Neural network prediction model for fine particulate matter (PM2.5) on the US–Mexico border in El Paso (Texas) and Ciudad Juárez (Chihuahua),Environmental Modelling & Software, Volume 20, Issue 5, 2005, Pages 547-559, ISSN 1364-8152
https://www.sciencedirect.com/science/article/abs/pii/S1364815204000830

[6] Chang, YS., Abimannan, S., Chiao, HT. et al. An ensemble learning based hybrid model and framework for air pollution forecasting. Environ Sci Pollut Res 27, 38155–38168 (2020).
https://pubmed.ncbi.nlm.nih.gov/32621183/

[7] Ping Jiang, Chen Li, Ranran Li, Hufang Yang,An innovative hybrid air pollution early-warning system based on pollutants forecasting and Extenics evaluation,Knowledge-Based Systems,Volume 164,2019, Pages 174-192,ISSN 0950-7051.
https://www.sciencedirect.com/science/article/abs/pii/S0950705118305288

[8] Yue-Shan Chang, Hsin-Ta Chiao, Satheesh Abimannan, Yo-Ping Huang, Yi-Ting Tsai, Kuan-Ming Lin,An LSTM-based aggregated model for air pollution forecasting,Atmospheric Pollution Research,Volume 11, Issue 8,2020,Pages 1451-1463,ISSN 1309-1042.
https://www.sciencedirect.com/science/article/pii/S1309104220301215

[9] Heydari, A., Majidi Nezhad, M., Astiaso Garcia, D. et al. Air pollution forecasting application based on deep learning model and optimization algorithm. Clean Techn Environ Policy 24, 607–621 (2022).
https://link.springer.com/article/10.1007/s10098-021-02080-5

[10] Adil Masood, Kafeel Ahmad, A review on emerging artificial intelligence (AI) techniques for air pollution forecasting: Fundamentals, application and performance, Journal of Cleaner Production, Volume 322, 2021, 129072, ISSN 0959-6526
https://sciencedirect.com/science/article/abs/pii/S0959652621032613

[11] A. Dairi, F. Harrou, S. Khadraoui and Y. Sun, "Integrated Multiple Directed Attention-Based Deep Learning for Improved Air Pollution Forecasting," in IEEE Transactions on Instrumentation and Measurement, vol. 70, pp. 1-15, 2021, Art no. 3520815, doi: 10.1109/TIM.2021.3091511.
https://ieeexplore.ieee.org/abstract/document/9466491

[12] Arsov, M.; Zdravevski, E.; Lameski, P.; Corizzo, R.; Koteli, N.; Gramatikov, S.; Mitreski, K.; Trajkovik, V. Multi-Horizon Air Pollution Forecasting with Deep Neural Networks. Sensors 2021, 21, 1235.
https://www.mdpi.com/1424-8220/21/4/1235/htm

[13] Yawei Dong, Chengyuan Zhang, Mingfei Niu, Shouyang Wang, Shaolong Sun, Air pollution forecasting with multivariate interval decomposition ensemble approach, Atmospheric Pollution Research, Volume 12, Issue 12, 2021, 101230, ISSN 1309-1042
https://www.sciencedirect.com/science/article/abs/pii/S1309104221002932

[14] Liao, K.; Huang, X.; Dang, H.; Ren, Y.; Zuo, S.; Duan, C. Statistical Approaches for Forecasting Primary Air Pollutants: A Review. Atmosphere 2021, 12, 686.
https://www.mdpi.com/2073-4433/12/6/686/htm

[15] Bingchun Liu, Xiaogang Yu, Jiali Chen, Qingshan Wang, Air pollution concentration forecasting based on wavelet transform and combined weighting forecasting model, Atmospheric Pollution Research, Volume 12, Issue 8, 2021, 101144, ISSN 1309-1042
https://www.sciencedirect.com/science/article/abs/pii/S1309104221002105

[16] K. Krishna Rani Samal, Ankit Kumar Panda, Korra Sathya Babu, Santos Kumar Das, Multi-output TCN autoencoder for long-term pollution forecasting for multiple sites,Urban Climate, Volume 39, 2021, 100943, ISSN 2212-0955
https://www.sciencedirect.com/science/article/abs/pii/S2212095521001735

[17] Leping Tu, Yan Chen, An unequal adjacent grey forecasting air pollution urban model, Applied Mathematical Modelling, Volume 99, 2021, Pages 260-275, ISSN 0307-904X
https://www.sciencedirect.com/science/article/abs/pii/S0307904X21003097

[18] González-Enrique, J.; Ruiz-Aguilar, J.J.; Moscoso-López, J.A.; Urda, D.; Deka, L.; Turias, I.J. Artificial Neural Networks, Sequence-to-Sequence LSTMs, and Exogenous Variables as Analytical Tools for NO2 (Air Pollution) Forecasting: A Case Study in the Bay of Algeciras (Spain). Sensors 2021, 21, 1770.
https://www.mdpi.com/1424-8220/21/5/1770/htm

[19] Bekkar, A., Hssina, B., Douzi, S. et al. Air-pollution prediction in smart city, deep learning approach. J Big Data 8, 161 (2021).
https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00548-1

[20] P. S. G. De Mattos Neto et al., "Neural-Based Ensembles for Particulate Matter Forecasting," in IEEE Access, vol. 9, pp. 14470-14490, 2021.
https://ieeexplore.ieee.org/abstract/document/9319268

[21] Xu, X.; Yang, H.; Li, C. Theoretical Model and Actual Characteristics of Air Pollution Affecting Health Cost: A Review. Int. J. Environ. Res. Public Health 2022, 19, 3532.
https://www.mdpi.com/1660-4601/19/6/3532/htm

[22] Nazar, W.; Niedoszytko, M. Air Pollution in Poland: A 2022 Narrative Review with Focus on Respiratory Diseases. Int. J. Environ. Res. Public Health 2022, 19, 895.
https://www.mdpi.com/1660-4601/19/2/895/htm

[23] Shuai Chen, Paulina Oliva, Peng Zhang, The effect of air pollution on migration: Evidence from China, Journal of Development Economics, Volume 156, 2022, 102833, ISSN 0304-3878
https://www.sciencedirect.com/science/article/abs/pii/S0304387822000153

[24] Leslie Edwards, Paul Wilkinson, Gemma Rutter, Ai Milojevic, Health effects in people relocating between environments of differing ambient air pollution concentrations: A literature review, Environmental Pollution, Volume 292, Part A, 2022, 118314, ISSN 0269-7491
https://www.sciencedirect.com/science/article/pii/S0269749121018960

[25] Xu Bai, Hui Chen, Brian G. Oliver, The health effects of traffic-related air pollution: A review focused the health effects of going green, Chemosphere, Volume 289, 2022, 133082, ISSN 0045-6535
https://www.sciencedirect.com/science/article/abs/pii/S0045653521035542

[26] Jayatra Mandal, Abhra Chanda, Sourav Samanta, Air pollution in three megacities of India during the Diwali festival amidst COVID-19 pandemic, Sustainable Cities and Society, Volume 76, 2022, 103504, ISSN 2210-6707
https://www.sciencedirect.com/science/article/abs/pii/S2210670721007708

[27] Long, E., Carlsten, C. Controlled human exposure to diesel exhaust: results illuminate health effects of traffic-related air pollution and inform future directions. Part Fibre Toxicol 19, 11 (2022)
https://particleandfibretoxicology.biomedcentral.com/articles/10.1186/s12989-022-00450-5

[28] Long, E., Schwartz, C. & Carlsten, C. Controlled human exposure to diesel exhaust: a method for understanding health effects of traffic-related air pollution. Part Fibre Toxicol 19, 15 (2022).
https://particleandfibretoxicology.biomedcentral.com/articles/10.1186/s12989-022-00454-1

[29] Stewart, C., Damby, D.E., Horwell, C.J. et al. Volcanic air pollution and human health: recent advances and future directions. Bull Volcanol 84, 11 (2022).
https://link.springer.com/article/10.1007/s00445-021-01513-9

[30] Ali Agga, Ahmed Abbou, Moussa Labbadi, Yassine El Houm, Imane Hammou Ou Ali, CNN-LSTM: An efficient hybrid deep learning architecture for predicting short-term photovoltaic power production, Electric Power Systems Research, Volume 208, 2022, 107908, ISSN 0378-7796
https://www.sciencedirect.com/science/article/abs/pii/S0378779622001389

[31]Ankita, Shalli Rani, Ali Kashif Bashir, Adi Alhudhaif, Deepika Koundal, Emine Selda Gunduz, An efficient CNN-LSTM model for sentiment detection in #BlackLivesMatter, Expert Systems with Applications, Volume 193, 2022, 116256, ISSN 0957-4174
https://www.sciencedirect.com/science/article/abs/pii/S0957417421015657

[32] Ghimire, S.; Deo, R.C.; Wang, H.; Al-Musaylh, M.S.; Casillas-Pérez, D.; Salcedo-Sanz, S. Stacked LSTM Sequence-to-Sequence Autoencoder with Feature Selection for Daily Solar Radiation Prediction: A Review and New Modeling Results. Energies 2022, 15, 1061.
https://www.mdpi.com/1996-1073/15/3/1061/htm

[33] Durmuş, B. Infinite impulse response system identification using average differential evolution algorithm with local search. Neural Comput & Applic 34, 375–390 (2022).
https://link.springer.com/article/10.1007/s00521-021-06399-4

[34] Alavi, J., Ewees, A.A., Ansari, S. et al. A new insight for real-time wastewater quality prediction using hybridized kernel-based extreme learning machines with advanced optimization algorithms. Environ Sci Pollut Res 29, 20496–20516 (2022).
https://link.springer.com/article/10.1007/s11356-021-17190-2

[35] Ziqiang Li, Gouhei Tanaka, Multi-reservoir echo state networks with sequence resampling for nonlinear time-series prediction, Neurocomputing, Volume 467, 2022, Pages 115-129, ISSN 0925-2312
https://www.sciencedirect.com/science/article/pii/S0925231221013333

[36] Myron Papadimitrakis, Alex Alexandridis, Active vehicle suspension control using road preview model predictive control and radial basis function networks, Applied Soft Computing, Volume 120, 2022, 108646, ISSN 1568-4946
https://www.sciencedirect.com/science/article/abs/pii/S1568494622001296

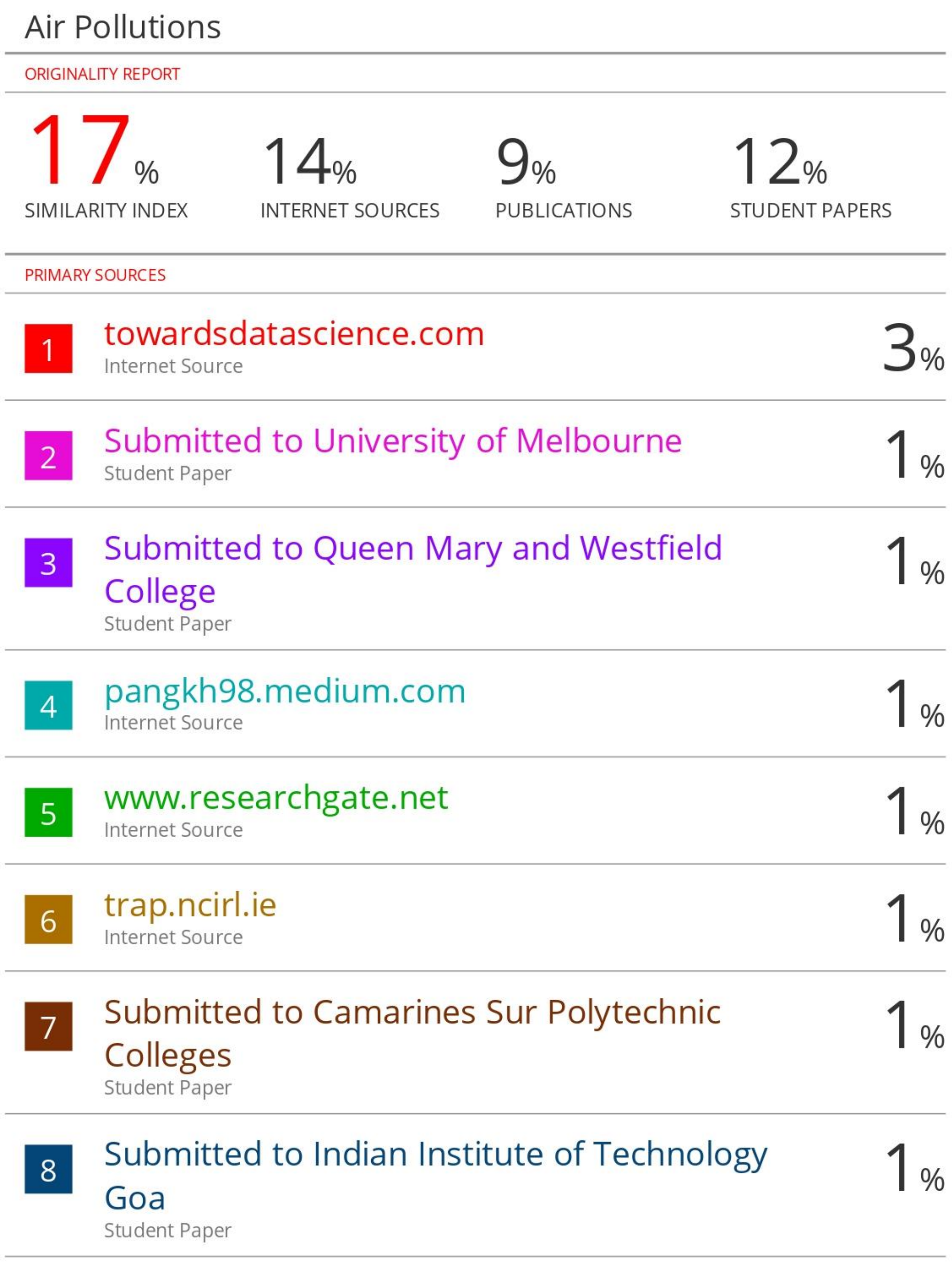
[37] Yang Han, Jacqueline CK Lam, Victor OK Li, David Reiner, A Bayesian LSTM model to evaluate the effects of air pollution control regulations in Beijing, China, Environmental Science & Policy, Volume 115, 2021, Pages 26-34, ISSN 1462-9011
https://www.sciencedirect.com/science/article/abs/pii/S1462901120313538

[38] Luke Conibear, Carly L. Reddington, Ben J. Silver, Ying Chen, Stephen R. Arnold, Dominick V. Spracklen, Emission Sector Impacts on Air Quality and Public Health in China From 2010 to 2020, GeoHealth, 10.1029/2021GH000567, 6, 6, (2022).
https://agupubs.onlinelibrary.wiley.com/doi/full/10.1029/2021GH000570

[39] Guoyan Huang, Xinyi Li, Bing Zhang, Jiadong Ren, PM2.5 concentration forecasting at surface monitoring sites using GRU neural network based on empirical mode decomposition, Science of The Total Environment, Volume 768, 2021, 144516, ISSN 0048-9697
https://www.sciencedirect.com/science/article/abs/pii/S0048969720380475

[40] H. Guo, Q. Chen, K. Zheng, Q. Xia and C. Kang, "Forecast Aggregated Supply Curves in Power Markets Based On LSTM Model," in IEEE Transactions on Power Systems, vol. 36, no. 6, pp. 5767-5779, Nov. 2021, doi: 10.1109/TPWRS.2021.3079923.
https://ieeexplore.ieee.org/abstract/document/9430706

[41] Xi Gao, Weide Li, A graph-based LSTM model for PM2.5 forecasting, Atmospheric Pollution Research, Volume 12, Issue 9, 2021, 101150, ISSN 1309-1042
https://www.sciencedirect.com/science/article/abs/pii/S1309104221002166



Air Pollutions

ORIGINALITY REPORT

17%	14%	9%	12%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	towardsdatascience.com Internet Source	3%
2	Submitted to University of Melbourne Student Paper	1%
3	Submitted to Queen Mary and Westfield College Student Paper	1%
4	pangkh98.medium.com Internet Source	1%
5	www.researchgate.net Internet Source	1%
6	trap.ncirl.ie Internet Source	1%
7	Submitted to Camarines Sur Polytechnic Colleges Student Paper	1%
8	Submitted to Indian Institute of Technology Goa Student Paper	1%