COMPARATIVE ANALYSIS OF VARIOUS MACHINE LEARNING, DEEP LEARNING AND HYBRID MODELS FOR AIR QUALITY PREDICTION

Report Submitted to the SASTRA Deemed to be University

as the Requirement of the course

CSE300 MINI PROJECT

Submitted by

Janupalli Saketh Reddy

(Reg No: 225003126)

Thumukunta Vamshi Karthikeya

(Reg No: 225003183)

M Krishna Kaushal

(Reg No: 225003199)

MAY 2024

Department of Computer Science and Engineering

SRINIVASA RAMANUJAN CENTRE

KUMBAKONAM, TAMIL NADU, INDIA – 612001

Department of Computer Science and Engineering

SRINIVASA RAMANUJAN CENTRE

KUMBAKONAM, TAMIL NADU, INDIA – 612001

Bonafide Certificate

This is to certify that the report titled **"Comparative Analysis of various Machine Learning, Deep Learning And Hybrid Models For Air Quality Prediction"** submitted as a requirement for the course, **CSE300**: **MINI PROJECT** for B.Tech. is a bonafide record of the work done by **Mr. JANUPALLI SAKETH REDDY(Reg. No.225003126), Mr. THUMUKUNTA VAMSHI KARTHIKEYA (Reg. No.225003183), Mr. M KRISHNA KAUSHAL (Reg. No.225003199)** during the academic year 2023-24, in the Srinivasa Ramanujan Centre, under my supervision.

Signature of Project Supervisor :

Name with Affiliation : **Mrs. Vanitha M/AP-II/CSE/SRC/SASTRA**

Date :

Mini Project *Viva voce* held on_____

Examiner 1 **Examiner 2**

ACKNOWLEDGEMENTS

We pay our sincere obeisance to the God Almighty for his grace and infinite mercy and for showing on us his choicest blessings.

We would like to express our thanks to our Chancellor **Prof. R. Sethuraman,** Vice Chancellor **Dr. S. Vaidhyasubramaniam** and Registrar **Dr. R. Chandramouli** for having given us an opportunity to be a student of this esteemed institution.

We express our deepest thanks to **Dr. V. Ramaswamy**, Dean and **Dr. A. Alli Rani**, Associate Dean, Srinivasa Ramanujan Centre for their constant support and suggestions when required without any reservations.

We express our gratitude to HOD in charge **Dr. V. Kalaichelvi**/ACP/CSE, Srinivasa Ramanujan Centre for her constant support and valuable suggestion for the completion of the project.

We exhibit our pleasure in expressing our thanks **Mrs. Vanitha M/AP-II/CSE**, our guide for her ever-encouraging spirit and meticulous guidance for the completion of the project.

We would like to place on record the benevolent approach and pain taking efforts of guidance and correction of **Dr. G Revathy APIII/CSE,** the project coordinator and all department staffs to whom we owe our hearty thanks for ever.

Without the support of our parents and friends this project would never have become reality.
We dedicate this work to our well-wishers, with love and affection.

List of Figures

List of Tables

Abbreviations

AQI – Air Quality Index

DL – Deep Learning

GRU - Gated Recurrent Unit

IQR – Inter Quartile Range

KNN – K Nearest Neighbor

LR – Linear Regression

LSTM – Long Short-Term Memory

MAE – Mean Absolute Error

ML – Machine Learning

MSE - Mean Square Error

R^2 – R-Squared Score

RMSE – Root Mean Square Error

SVM – Support Vector Machine

Notations

a_i = Estimated Intercept

AQI = The (Air Quality) Index

b = Bias

C = Pollutant Concentration

C_{low} = Concentration breakpoint <=c

C_{high} = Concentration breakpoint >=c

$c_t \in Rh$ = Vector to be used for cell state

d = Euclidean Distance

f(x) = Function with Least possible deviation

$f_t \in Rh$ = Activation vector used for Forget gate of LSTM unit

$h_t \in Rh$ = Output vector of LSTM unit. (Hidden State value)

I_{low} = Index breakpoint respective to C_{low}

I_{high} = Index breakpoint respective to C_{high}

$i_t \in Rh$ = Activation vector used for Input gate of LSTM unit

$o_t \in Rh$ = Activation vector used for Output gate of LSTM unit

T = Training dataset

w = Vector of weights

$W \in R^{(h*d)}$, $U \in R^{(h*h)}$, $b \in R^{(h)}$ = Weight matrices and bias vector parameters that are trainable where superscripts ∈ h and ∈ d refer to the number of input features and number of hidden units respectively.

X = Real Value input Vector

x_i = Input space

$x_t \in Rd$ = Vector to be used as input to LSTM unit

Y = Real Value output Vector

φ(xi) = High-Dimensional feature space

σ_g = Logarithmic Sigmoid function

σ_c = Hyperbolic tangent function

σ_h = Hyperbolic tangent function or identity function

Abstract

Concern over air pollution has grown to be a major issue affecting environmental sustainability and public health in today's fast urbanizing world. Accurate Air Quality Index (AQI) forecasting is becoming more and more important as cities grow and industrialization picks up speed, the goal of this project is to improve the sufficiency of current technologies in forecasting Air Quality Index (AQI) levels. Before relocating to any region that is filthy, the Air Quality Index (AQI) prediction can help them to take precautions regarding their health. Understanding how important AQI is for evaluating contaminants and directing regulatory actions, this project suggests a Hybrid Model for AQI prediction. The proposed model seeks to offer precise and up-to-date projections by utilising past air quality data, meteorological characteristics.

We can now analyze enormous volumes of data and discover intricate patterns and connections that conventional methods alone would not reveal, thanks to machine learning algorithms. Deep learning techniques further enhance predictive capabilities by automatically learning hierarchical representations of data. By combining these modern computational techniques, we can generate forecasts with unprecedented accuracy and gain valuable insights into the variables influencing air quality.

This project will aid in making educated decisions regarding environmental sustainability. In order to promote a cleaner and healthier future, the project is in line with the urge to enable communities and authorities to proactively battle air pollution.

Keywords: Air Quality Index, Air Pollution, Urbanization, Sustainability, Forecasting, Meteorological Characteristics.

Table of Contents

CHAPTER 1

SUMMARY OF THE BASE PAPER

Title: Air Quality Index prediction using an effective hybrid deep learning model

Journal: Environmental Pollution

Publisher: ELSEVIER

Year: 2022

Indexing: SCI-E

1.1 Summary:

- The paper emphasizes the critical importance of forecasting the Air Quality Index (AQI) to empower individuals in protecting their health, particularly in regions plagued by heightened pollution levels.
- It systematically examines a range of computational models, spanning both machine learning and deep learning methodologies, to evaluate their efficacy in AQI prediction.
- Following rigorous analysis, the study discerns that the proposed hybrid Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) model outperforms its standalone counterparts.
- This hybrid model demonstrates commendable performance metrics, including a Mean Absolute Error (MAE) of 36.11 and an impressive R-squared (R^2) coefficient of 0.84.
- Notably, the hybrid model's capability in AQI prediction is underscored by its significantly reduced error rates.
- These findings underscore the potential of advanced computational techniques to enhance the accuracy and reliability of AQI forecasts.
- Consequently, proactive interventions can be facilitated to mitigate the health risks posed by ambient air pollution, based on more precise AQI predictions.

1.2 Research Addressed:

- The research addresses the urgent necessity for precise Air Quality Index (AQI) forecasts in densely polluted urban areas, highlighting its profound implications for managing public health.
- By developing and evaluating a hybrid deep learning model (LSTM-GRU), the study seeks to refine existing predictive methods, comparing the hybrid model's performance with other commonly used machine learning and deep learning models for AQI prediction.

- Emphasis is placed on key metrics like Mean Absolute Error (MAE) and R-squared (R2) values, underscoring the significance of accuracy in AQI predictions to inform individuals about potential health hazards linked to air pollution.
- Additionally, the research underscores the vital role of predictive models in facilitating proactive measures to mitigate the adverse health consequences of poor air quality.
- Through demonstrating the hybrid model's superior accuracy and performance metrics, the study not only advances the realm of air quality prediction but also furnishes practical insights for policymakers, environmental agencies, and healthcare professionals to effectively manage and alleviate the impacts of air pollution on public health.

1.3 Proposed Solution:

- The proposed solution integrates LSTM (Long Short-Term Memory), GRU (Gated Recurrent Unit), and RNN (Recurrent Neural Network) architectures.
- LSTM networks excel in retaining information over extended sequences, crucial for capturing long-term patterns in AQI (Air Quality Index) data.
- GRU networks offer computational efficiency compared to LSTM, while still effectively capturing short-term dependencies.
- RNNs provide a flexible framework for processing sequential data.
- By combining LSTM, GRU and RNN, the hybrid model leverages their complementary strengths, enhancing both short-term fluctuations and long-term patterns in AQI data.
- The hybrid model undergoes training using historical AQI data, optimizing its parameters to minimize prediction errors through iterative adjustments.
- During evaluation, the hybrid model demonstrates superior performance compared to standalone LSTM, GRU and RNN models, as well as traditional ML methods such as linear regression, support vector machines (SVM), and k-nearest neighbors (KNN).
- The integration of LSTM, GRU, and RNN allows the model to capture both short-term fluctuations and long-term trends in AQI data more effectively, leading to improved prediction accuracy.

1.4 Proposed Architecture:

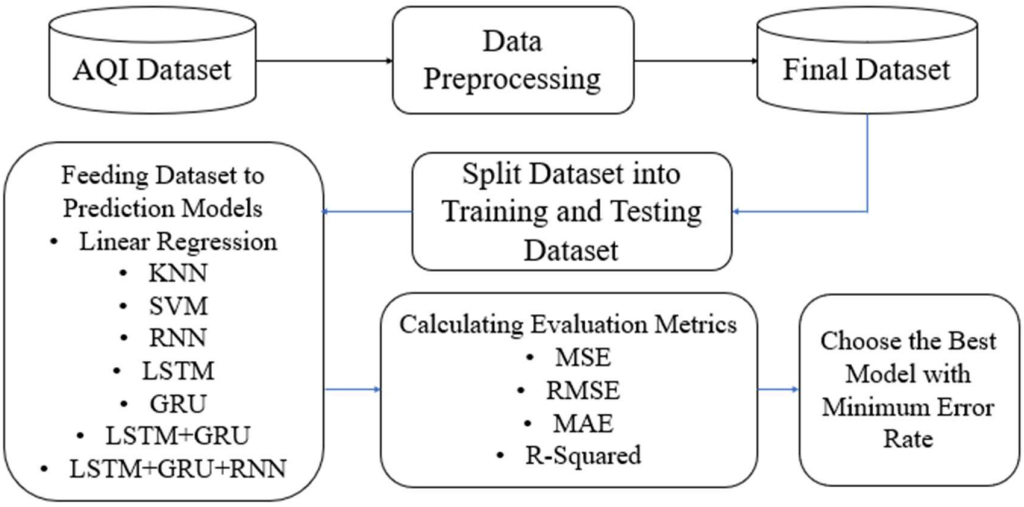


Fig. 1. Architecture Diagram

1.5 Prediction Models Used:

1.5.1 Linear Regression (LR):

- Utilized for its simplicity and interpretability in modeling linear relationships between input features and AQI values.
- Predicts AQI levels based on a weighted sum of input variables, assuming a linear relationship between predictors and AQI.

Formula: $y = a_0 + \sum_{i=1}^{m} a_i x_i$

1.5.2 Support Vector Machines (SVM):

- Employed to find the optimal hyperplane that separates different AQI classes.
- Classifies AQI levels based on their proximity to the decision boundary defined by the hyperplane.

Formula: $f(x) = w^T \varphi(x_i) + b$

1.5.3 K-Nearest Neighbors (KNN):

- Predicts the AQI of a given location based on the majority class of its k nearest neighbors.
- Utilizes the proximity of data points in the feature space to determine AQI levels.

Formula: $d = \sum_{i=0}^{n} \sqrt{(a_i - b_i)^2}$

1.5.4 Recurrent Neural Network (RNN):

- A class of neural networks specifically designed to process sequential data by retaining information in memory over time.
- Suitable for tasks such as time series forecasting, where capturing temporal dependencies is crucial.

Formula:

$$\boldsymbol{a}^{(t)} = b + \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)}$$
$$\boldsymbol{h}^{(t)} = tanh(\boldsymbol{a}^{(t)})$$
$$\boldsymbol{o}^{(t)} = c + \boldsymbol{V}\boldsymbol{h}^{(t)}$$
$$\widehat{\boldsymbol{y}}^{(t)} = \sigma(\boldsymbol{o}^{(t)})$$

$$\sum_{t=1}^{\tau} L^{(t)} = L\Big((\boldsymbol{x}^{(1)}, \ldots\ldots, \boldsymbol{x}^{(\tau)}), (\boldsymbol{y}^{(1)}, \ldots\ldots, \boldsymbol{y}^{(\tau)})\Big)$$

1.5.5 Long Short-Term Memory (LSTM) Network:

- A type of recurrent neural network (RNN) known for its ability to capture long-term dependencies in sequential data.
- Effective in learning and remembering patterns over extended periods, making it suitable for modeling the complex dynamics of air quality changes.

Formula:

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i)$$
$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f)$$
$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o)$$
$$c_t = f_t \circ c_{(t-1)} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c)$$
$$h_t = o_t \circ \sigma_h(c_t)$$

1.5.6 Gated Recurrent Unit (GRU) Network:

- Another type of recurrent neural network designed to address the vanishing gradient problem and improve training efficiency.
- Simplifies the architecture by merging the memory and input gates, leading to faster training and inference times compared to LSTM.

Formula:

$$r_t = \sigma(W_r.[h_{t-1}, x_t])$$

$$z_t = \sigma(W_z.[h_{t-1}, x_t])$$

$$h_t = \tanh(W_h.[r_t * h_{t-1}, x_t])$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * h_t$$

$$y_t = \sigma(W_o.h_t)$$

1.5.7 Hybrid LSTM-GRU Model:

- Integrates the strengths of both LSTM and GRU architectures for enhanced AQI forecasting.
- Combines LSTM's proficiency in capturing long-term dependencies with GRU's computational efficiency and effective management of short-term information.
- Offers a robust solution capable of discerning both short-term fluctuations and long-term patterns in AQI data.
- Outperforms standalone models and traditional machine learning algorithms, providing accurate forecasts for air quality management and public health initiatives in Delhi.

1.5.8 LSTM+GRU+RNN Model:

- Comprehensive hybrid model combining LSTM, GRU, and Recurrent Neural Network (RNN) architectures.
- Incorporates LSTM's long-term memory retention, GRU's computational efficiency, and RNN's flexibility.
- Excels in capturing complex temporal dependencies, making it well-suited for tasks such as air quality forecasting where both short-term and long-term patterns are critical.
- Offers a robust solution capable of discerning both short-term fluctuations and long-term patterns in AQI data.
- Outperforms standalone models and traditional machine learning algorithms, providing accurate forecasts for air quality management and public health initiatives in Delhi.

1.5.9 Definition:

AQI Category	**PM_{10}**	**$PM_{2.5}$**	**NO_2**	**O_3**	**CO**	**SO_2**	**NH_3**	**Pb**
Good (0-50)	0-50	0-30	0-40	0-50	0-1.0	0-40	0-200	0-0.5
Satisfactory (51-100)	51-100	31-60	41-80	51-100	1.1-2.0	41-80	201-400	0.5-1.0
Moderate (101-200)	101-250	61-90	81-180	101-168	2.1-10	18-380	401-800	1.1-2.0
Poor (201-300)	251-300	91-120	181-280	169-280	10-17	381-800	801-1200	2.1-3.0
Very Poor (301-400)	351-430	120-250	281-400	209-748	17-34	801-1600	1200-1800	3.1-3.5
Severe (401-500)	430+	250+	400+	748+	34+	1600+	1800+	3.5+

Table 1: Breakpoints

AQI definitions

AQI	Associated health impacts
Good (0–50)	Minimal impact
Satisfactory (51–100)	May cause minor breathing discomfort to sensitive people.
Moderate (101–200)	May cause breathing discomfort to people with lung disease such as asthma, and discomfort to people with heart disease, children and older adults.
Poor (201–300)	May cause breathing discomfort to people on prolonged exposure, and discomfort to people with heart disease.
Very Poor (301–400)	May cause respiratory illness to the people on prolonged exposure. Effect may be more pronounced in people with lung and heart diseases.
Severe (401–500)	May cause respiratory impact even on healthy people, and serious health impacts on people with lung/heart disease. The health impacts may be experienced even during light physical activity.

Table 2: AQI Definition

1.5.10 Formula:

$$I = \frac{I_{high} - I_{low}}{C_{high} - C_{low}}(C - C_{low}) + I_{low}$$

Fig 2: Formula

AQI Category	PM_{10}	$PM_{2.5}$	NO_2	O_3	CO	SO_2	Pb
0-50	0-54	0-9	0-53	0-50	0-4.4	0-35	0-0.5
51-100	55-154	9.1-35.4	54-100	51-125	4.5-9.4	36-75	0.5-1.0
101-200	155-254	35.5-55.4	101-360	125-164	9.5-12.4	76-185	1.1-2.0
201-300	255-354	55.5-125.4	361-649	165-204	12.5-15.4	186-304	2.1-3.0
301-400	355-424	125.5-225.4	650-1249	205-404	15.5-30.4	305-604	3.1-3.5
401-500	425-604	225.5-325.4	1250-2049	405-604	30.5-50.4	605-1004	3.5-4.4

Table 3: Parameter Value Range

CHAPTER 2

MERITS AND DEMERITS OF THE BASE PAPER

2.1 MERITS

- This base paper includes the development of a hybrid LSTM-GRU model for accurate prediction of the Air Quality Index (AQI).
- Both LSTM and GRU architectures are designed to address the vanishing gradient problem encountered in traditional RNNs, allowing them to capture long-term dependencies in sequential data.
- Memory cell processes in LSTM and GRU designs let them to selectively remember or forget information over time. This feature helps in AQI prediction by enabling the model to retain significant historical observations while removing unimportant noise.
- Compared to linear models like Linear Regression and SVM, LSTM and GRU networks are better able to capture these intricate interactions because of their capacity to simulate nonlinearities through recurrent connections and memory cells. Although KNN can capture nonlinear relationships.
- The hybrid LSTM-GRU model outperforms standalone ML and DL models in terms of Mean Absolute Error (MAE) and R-squared (R2) values.
- This hybrid model shows superior performance with a low MAE value of 36.11, indicating its accuracy in predicting the Air Quality Index.
- The R2 value of the hybrid model is 0.84, demonstrating a strong correlation between the predicted and actual AQI values.
- The hybrid model mentioned in base paper shows supremacy in performance compared to LSTM, Linear Regression, GRU, K-Nearest Neighbor, and Support Vector Machine models.
- This model considers pollutant content and meteorological characteristics, highlighting the significance of AQI prediction.
- The proposed model provides a more accurate prediction of AQI values, essential for public health and environmental sustainability.

2.2 DEMERITS

- The study is restricted to a single city, and other data sources would be needed to extend the prediction of AQI to other locations.
- This paper ignores the accuracy of machine learning and deep learning model while mainly focusing on proposed model. As the input data is limited, the accuracy of the model will not be high.
- Any conflicting financial interests or personal connections that might have impacted the work are not included in the report.
- LSTM and GRU architectures are more complex models.
- The study does not provide a detailed discussion of potential biases or limitations in the methodology used for AQI prediction.

CHAPTER 3

SOURCE CODE

3.1 Preprocessing

3.1.1 Step-1

```
import pandas as pd
import numpy as np
import time

# Data Visualization Libraries
import missingno as msno
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Read data
data = pd.read_csv(r'C:\Users\kaush\Desktop\MINI PROJECT\step1.csv')
# Display first few rows of the data
data.head()
# Check for missing values
missing_values = data.isnull().sum()
print("Total missing data =", missing_values.sum())
# Calculate missing percentage
missing_percentage = (missing_values.sum() / (data.size)) * 100
print("Total missing percentage =", missing_percentage)
```

```
# Select numerical columns
numerical_cols = data.select_dtypes(include='number')
# Fill missing values with median
start_time = time.time()
filled_data = numerical_cols.fillna(numerical_cols.median())
end_time = time.time()
elapsed_time = end_time - start_time
print("Time spent for median imputation =", elapsed_time)
# Add date column to filled data
filled_data.insert(0, "Date", data["Date"])
# Save filled data to a new file
filled_data.to_csv(r"C:\Users\kaush\Desktop\MINI PROJECT\step1_imp_median.csv", index=False)
```

3.1.2 Step-2

```
import pandas as pd
import numpy as np
import random
from sklearn import preprocessing
import missingno as msno

import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Calculate IQR for PM2.5
data['PM25'] = data['PM25'].apply(pd.to_numeric, errors='coerce')
Q1_PM25 = data['PM25'].quantile(0.25)
Q3_PM25 = data['PM25'].quantile(0.75)
IQR_PM25 = Q3_PM25 - Q1_PM25

# Calculate IQR for PM10
data['PM10'] = data['PM10'].apply(pd.to_numeric, errors='coerce')
Q1_PM10 = data['PM10'].quantile(0.25)
Q3_PM10 = data['PM10'].quantile(0.75)
IQR_PM10 = Q3_PM10 - Q1_PM10

# Calculate IQR for CO
data['CO'] = data['CO'].apply(pd.to_numeric, errors='coerce')
Q1_CO = data['CO'].quantile(0.25)
Q3_CO = data['CO'].quantile(0.75)
IQR_CO = Q3_CO - Q1_CO

# Calculate IQR for NH3
data['NH3'] = data['NH3'].apply(pd.to_numeric, errors='coerce')
Q1_NH3 = data['NH3'].quantile(0.25)
Q3_NH3 = data['NH3'].quantile(0.75)
IQR_NH3 = Q3_NH3 - Q1_NH3

# Calculate IQR for OZONE
data['OZONE'] = data['OZONE'].apply(pd.to_numeric, errors='coerce')
Q1_OZONE = data['OZONE'].quantile(0.25)
Q3_OZONE = data['OZONE'].quantile(0.75)
IQR_OZONE = Q3_OZONE - Q1_OZONE

# Calculate IQR for SO2
data['SO2'] = data['SO2'].apply(pd.to_numeric, errors='coerce')
Q1_SO2 = data['SO2'].quantile(0.25)
Q3_SO2 = data['SO2'].quantile(0.75)
IQR_SO2 = Q3_SO2 - Q1_SO2
```

```
data['Season'] = pd.cut(data['Month'], bins=[0, 3, 6, 9, 12], labels=['Spring', 'Summer', 'Fall', 'Winter'])

# Step 4: Assign weekday or not
data['Is_Weekday'] = data['Date'].dt.dayofweek.isin([0, 1, 2, 3, 4])

# Step 5: Change Date column from datetime64[ns] to Object back after weekday calculation
data['Date'] = data['Date'].dt.strftime('%Y-%m-%d')
```

```
# Store file as AV2015_2023_step1.csv
data.to_csv("AV2015_2023_step1.csv", index=False)
```

3.1.3 Step-3

```
import pandas as pd
import numpy as np
```

```
# Load your dataset here (replace 'your_dataset.csv' with your actual file path)
data = pd.read_csv('your_dataset.csv')

# Define AQI breakpoints and corresponding index values for PM2.5
aqi_breakpoints_pm25 = [(0, 12), (12.1, 35.4), (35.5, 55.4), (55.5, 150.4), (150.5, 250.4), (250.5, 350.4), (350.5, 500.4)]
aqi_values_pm25 = [0, 51, 101, 151, 201, 301, 401, 500]

# Function to calculate AQI for PM2.5
def calculate_aqi_pm25(pm25):
    for i, (low, high) in enumerate(aqi_breakpoints_pm25):
        if low <= pm25 <= high:
            aqi = ((aqi_values_pm25[i + 1] - aqi_values_pm25[i]) / (high - low)) * (pm25 - low) + aqi_values_pm25[i]
            return round(aqi)

# Calculate AQI for PM2.5 column
data['AQI_PM25'] = data['PM25'].apply(calculate_aqi_pm25)

# Define AQI breakpoints and corresponding index values for other parameters
aqi_breakpoints_pm10 = [(0, 54), (55, 154), (155, 254), (255, 354), (355, 424), (425, 504), (505, 604), (605, 1004)]
aqi_values_pm10 = [0, 51, 101, 151, 201, 301, 401, 501, 601, 701]

# Function to calculate AQI for PM10
def calculate_aqi_pm10(pm10):
    for i, (low, high) in enumerate(aqi_breakpoints_pm10):
        if low <= pm10 <= high:
            aqi = ((aqi_values_pm10[i + 1] - aqi_values_pm10[i]) / (high - low)) * (pm10 - low) + aqi_values_pm10[i]
            return round(aqi)

# Calculate AQI for PM10 column
data['AQI_PM10'] = data['PM10'].apply(calculate_aqi_pm10)
```

```
# Define AQI breakpoints and corresponding index values for CO
aqi_breakpoints_co = [(0, 4.4), (4.5, 9.4), (9.5, 12.4), (12.5, 15.4), (15.5, 30.4), (30.5, 40.4), (40.5, 50.4), (50.5, 60.4)]
aqi_values_co = [0, 51, 101, 151, 201, 301, 401, 501, 601]

# Function to calculate AQI for CO
def calculate_aqi_co(co):
    for i, (low, high) in enumerate(aqi_breakpoints_co):
        if low <= co <= high:
            aqi = ((aqi_values_co[i + 1] - aqi_values_co[i]) / (high - low)) * (co - low) + aqi_values_co[i]
            return round(aqi)

# Calculate AQI for CO column
data['AQI_CO'] = data['CO'].apply(calculate_aqi_co)

# Define AQI breakpoints and corresponding index values for OZONE
aqi_breakpoints_ozone = [(0, 54), (55, 70), (71, 85), (86, 105), (106, 200), (201, 504)]
aqi_values_ozone = [0, 51, 101, 151, 201, 301, 401]

# Function to calculate AQI for OZONE
def calculate_aqi_ozone(ozone):
    for i, (low, high) in enumerate(aqi_breakpoints_ozone):
        if low <= ozone <= high:
            aqi = ((aqi_values_ozone[i + 1] - aqi_values_ozone[i]) / (high - low)) * (ozone - low) + aqi_values_ozone[i]
            return round(aqi)

# Calculate AQI for OZONE column
data['AQI_OZONE'] = data['OZONE'].apply(calculate_aqi_ozone)

# Define AQI breakpoints and corresponding index values for SO2
aqi_breakpoints_so2 = [(0, 35), (36, 75), (76, 185), (186, 304), (305, 604), (605, 804), (805, 1004), (1005, 1204)]
aqi_values_so2 = [0, 51, 101, 151, 201, 301, 401, 501, 601]
```

```
# Function to calculate AQI for SO2
def calculate_aqi_so2(so2):
    for i, (low, high) in enumerate(aqi_breakpoints_so2):
        if low <= so2 <= high:
            aqi = ((aqi_values_so2[i + 1] - aqi_values_so2[i]) / (high - low)) * (so2 - low) + aqi_values_so2[i]
            return round(aqi)

# Calculate AQI for SO2 column
data['AQI_SO2'] = data['SO2'].apply(calculate_aqi_so2)

# Define AQI breakpoints and corresponding index values for NH3
aqi_breakpoints_nh3 = [(0, 54), (55, 104), (105, 154), (155, 204), (205, 304), (305, 604), (605, 804)]
aqi_values_nh3 = [0, 51, 101, 151, 201, 301, 401, 501]
```

```
# Function to calculate AQI for NH3
def calculate_aqi_nh3(nh3):
    for i, (low, high) in enumerate(aqi_breakpoints_nh3):
        if low <= nh3 <= high:
            aqi = ((aqi_values_nh3[i + 1] - aqi_values_nh3[i]) / (high - low)) * (nh3 - low) + aqi_values_nh3[i]
            return round(aqi)

# Calculate AQI for NH3 column
data['AQI_NH3'] = data['NH3'].apply(calculate_aqi_nh3)

# Determine which parameter has the maximum AQI value
parameters = ['PM25', 'PM10', 'CO', 'OZONE', 'SO2', 'NH3']
data['Max_AQI_Parameter'] = data[parameters].apply(max, axis=1).idxmax()

# Store file as AV2015_2023_step1
data.to_csv("AV2015_2023_step3.csv", index=False)
```

3.2 Machine Learning Models

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import matplotlib.pyplot as plt
```

```
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split, KFold
from sklearn.pipeline import Pipeline
```

```
air_quality_data = pd.read_csv(r'C:\Users\tvams\OneDrive\Desktop\Imputation methods\Step-3.csv')
```

```
scaler = MinMaxScaler()
normalized_data = scaler.fit_transform(air_quality_data.drop(columns=['Date','AQI']))
X_normalized = normalized_data
y = air_quality_data['AQI']
```

```
X_train, X_test, y_train, y_test = train_test_split(X_normalized, y, test_size=0.2, random_state=42)
```

```
# Linear Regression
print("Linear Regression:")
model_lr = LinearRegression()
model_lr.fit(X_train, y_train)
y_pred_lr = model_lr.predict(X_test)
mse_lr = mean_squared_error(y_test, y_pred_lr)
mae_lr = mean_absolute_error(y_test, y_pred_lr)
rmse_lr = np.sqrt(mse_lr)
r2_lr = r2_score(y_test, y_pred_lr)
print("Mean Squared Error:", mse_lr)
print("Mean Absolute Error:", mae_lr)
print("Root Mean Squared Error:", rmse_lr)
print("R-squared:", r2_lr)
print()
```

```
# KNN
print("K-Nearest Neighbors:")
model_knn = KNeighborsRegressor(n_neighbors=5)
model_knn.fit(X_train, y_train)
y_pred_knn = model_knn.predict(X_test)
mse_knn = mean_squared_error(y_test, y_pred_knn)
mae_knn = mean_absolute_error(y_test, y_pred_knn)
rmse_knn = np.sqrt(mse_knn)
r2_knn = r2_score(y_test, y_pred_knn)
print("Mean Squared Error:", mse_knn)
print("Mean Absolute Error:", mae_knn)
print("Root Mean Squared Error:", rmse_knn)
print("R-squared:", r2_knn)
print()
```

```
# SVM
print("Support Vector Machine:")
model_svm = SVR(kernel='rbf')
model_svm.fit(X_train, y_train)
y_pred_svm = model_svm.predict(X_test)
mse_svm = mean_squared_error(y_test, y_pred_svm)
mae_svm = mean_absolute_error(y_test, y_pred_svm)
rmse_svm = np.sqrt(mse_svm)
r2_svm = r2_score(y_test, y_pred_svm)
print("Mean Squared Error:", mse_svm)
print("Mean Absolute Error:", mae_svm)
print("Root Mean Squared Error:", rmse_svm)
print("R-squared:", r2_svm)
```

```
plt.scatter(y_test, y_pred_lr, color='blue')
plt.xlabel('Actual AQI')
plt.ylabel('Predicted AQI')
plt.title('Actual vs. Predicted AQI (Linear Regression)')
plt.show()

plt.scatter(y_test, y_pred_knn, color='blue')
plt.xlabel('Actual AQI')
plt.ylabel('Predicted AQI')
plt.title('Actual vs. Predicted AQI (Linear Regression)')
plt.show()

plt.scatter(y_test, y_pred_svm, color='blue')
plt.xlabel('Actual AQI')
plt.ylabel('Predicted AQI')
plt.title('Actual vs. Predicted AQI (Linear Regression)')
plt.show()
```

3.3 Deep Learning Models

```
[ ]: import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
```

```
[ ]: from sklearn.preprocessing import MinMaxScaler
     from sklearn.model_selection import train_test_split,KFold
     from tensorflow.keras.models import Sequential
     from tensorflow.keras.layers import LSTM, GRU, SimpleRNN, Dense
     from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
     import time
```

```
[ ]: # Load the air quality data
     air_quality_data = pd.read_csv('step1_imp_median_aqi.csv')
```

```
[ ]: # Perform min-max normalization
     scaler = MinMaxScaler()
     normalized_data = scaler.fit_transform(air_quality_data.drop(columns=['Date', 'AQI']))
     X_normalized = normalized_data
     y = air_quality_data['AQI']
```

```
[ ]: # Reshape data for LSTM/GRU/RNN input
     X_train, X_test, y_train, y_test = train_test_split(X_normalized, y, test_size=0.2, random_state=42)
     X_train = X_train.reshape(X_train.shape[0], 1, X_train.shape[1])
     X_test = X_test.reshape(X_test.shape[0], 1, X_test.shape[1])
```

```
# Define LSTM model
start_time = time.time()
lstm_model = Sequential([
    LSTM(50, activation='relu', input_shape=(X_train.shape[1], X_train.shape[2])),
    Dense(1)
])

# Compile LSTM model
lstm_model.compile(optimizer='adam', loss='mse')
end_time = time.time()
compilation_time_lstm = end_time - start_time

# Train LSTM model
lstm_history = lstm_model.fit(X_train, y_train, epochs=100, batch_size=32, validation_data=(X_test, y_test), verbose=1)

# Predict using LSTM model
lstm_y_pred = lstm_model.predict(X_test)

# Calculate evaluation metrics for LSTM model
lstm_mae = mean_absolute_error(y_test, lstm_y_pred)
lstm_mse = mean_squared_error(y_test, lstm_y_pred)
lstm_rmse = np.sqrt(lstm_mse)
lstm_r2 = r2_score(y_test, lstm_y_pred)

print("LSTM Model Evaluation:")
print(f"Compilation Time: {compilation_time_lstm:.2f} seconds")
print(f"MAE: {lstm_mae:.4f}")
print(f"MSE: {lstm_mse:.4f}")
print(f"RMSE: {lstm_rmse:.4f}")
print(f"R2 Score: {lstm_r2:.4f}")
```

```
# Define GRU model
start_time = time.time()
gru_model = Sequential([
    GRU(50, activation='relu', input_shape=(X_train.shape[1], X_train.shape[2])),
    Dense(1)
])

# Compile GRU model
gru_model.compile(optimizer='adam', loss='mse')
end_time = time.time()
compilation_time_gru = end_time - start_time

# Train GRU model
gru_history = gru_model.fit(X_train, y_train, epochs=100, batch_size=32, validation_data=(X_test, y_test), verbose=1)

# Predict using GRU model
gru_y_pred = gru_model.predict(X_test)

# Calculate evaluation metrics for GRU model
gru_mae = mean_absolute_error(y_test, gru_y_pred)
gru_mse = mean_squared_error(y_test, gru_y_pred)
gru_rmse = np.sqrt(gru_mse)
gru_r2 = r2_score(y_test, gru_y_pred)

print("\nGRU Model Evaluation:")
print(f"Compilation Time: {compilation_time_gru:.2f} seconds")
print(f"MAE: {gru_mae:.4f}")
print(f"MSE: {gru_mse:.4f}")
print(f"RMSE: {gru_rmse:.4f}")
print(f"R2 Score: {gru_r2:.4f}")
```

```
# Define RNN model
start_time = time.time()
rnn_model = Sequential([
    SimpleRNN(50, activation='relu', input_shape=(X_train.shape[1], X_train.shape[2])),
    Dense(1)
])

# Compile RNN model
rnn_model.compile(optimizer='adam', loss='mse')
end_time = time.time()
compilation_time_rnn = end_time - start_time

# Train RNN model
rnn_history = rnn_model.fit(X_train, y_train, epochs=100, batch_size=32, validation_data=(X_test, y_test), verbose=1)

# Predict using RNN model
rnn_y_pred = rnn_model.predict(X_test)

# Calculate evaluation metrics for RNN model
rnn_mae = mean_absolute_error(y_test, rnn_y_pred)
rnn_mse = mean_squared_error(y_test, rnn_y_pred)
rnn_rmse = np.sqrt(rnn_mse)
rnn_r2 = r2_score(y_test, rnn_y_pred)

print("\nRNN Model Evaluation:")
print(f"Compilation Time: {compilation_time_rnn:.2f} seconds")
print(f"MAE: {rnn_mae:.4f}")
print(f"MSE: {rnn_mse:.4f}")
print(f"RMSE: {rnn_rmse:.4f}")
print(f"R2 Score: {rnn_r2:.4f}")
```

```
# Plot training history for LSTM, GRU, and RNN models
plt.figure(figsize=(10, 6))
plt.plot(lstm_history.history['loss'], label='LSTM Training Loss', color='blue')
plt.plot(lstm_history.history['val_loss'], label='LSTM Validation Loss', color='orange')
plt.plot(gru_history.history['loss'], label='GRU Training Loss', color='green')
plt.plot(gru_history.history['val_loss'], label='GRU Validation Loss', color='red')
plt.plot(rnn_history.history['loss'], label='RNN Training Loss', color='purple')
plt.plot(rnn_history.history['val_loss'], label='RNN Validation Loss', color='brown')
plt.title('Model Training History')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```

```
# Plot actual vs predicted AQI values for LSTM model
plt.figure(figsize=(10, 6))
plt.scatter(y_test, lstm_y_pred, color='blue')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=2)
plt.xlabel('Actual AQI')
plt.ylabel('Predicted AQI')
plt.title('Actual vs Predicted AQI (LSTM)')
plt.tight_layout()
plt.show()
# Plot actual vs predicted AQI values for GRU model
plt.figure(figsize=(10, 6))
plt.scatter(y_test, gru_y_pred, color='red')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=2)
plt.xlabel('Actual AQI')
plt.ylabel('Predicted AQI')
plt.title('Actual vs Predicted AQI (GRU)')
plt.tight_layout()
plt.show()

# Plot actual vs predicted AQI values for RNN model
plt.figure(figsize=(10, 6))
plt.scatter(y_test, rnn_y_pred, color='purple')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=2)
plt.xlabel('Actual AQI')
plt.ylabel('Predicted AQI')
plt.title('Actual vs Predicted AQI (RNN)')
plt.tight_layout()
plt.show()
```

```
# Plot actual vs predicted AQI values for LSTM model
plt.figure(figsize=(10, 6))
plt.plot(y_test.values, label='Actual', color='blue')
plt.plot(lstm_y_pred, label='Predicted', color='red')
plt.xlabel('Index')
plt.ylabel('AQI')
plt.title('Actual vs Predicted AQI (LSTM)')
plt.legend()
plt.tight_layout()
plt.show()

# Plot actual vs predicted AQI values for GRU model
plt.figure(figsize=(10, 6))
plt.plot(y_test.values, label='Actual', color='blue')
plt.plot(gru_y_pred, label='Predicted', color='red')
plt.xlabel('Index')
plt.ylabel('AQI')
plt.title('Actual vs Predicted AQI (GRU)')
plt.legend()
plt.tight_layout()
plt.show()

# Plot actual vs predicted AQI values for GRU model
plt.figure(figsize=(10, 6))
plt.plot(y_test.values, label='Actual', color='blue')
plt.plot(rnn_y_pred, label='Predicted', color='red')
plt.xlabel('Index')
plt.ylabel('AQI')
plt.title('Actual vs Predicted AQI (RNN)')
plt.legend()
plt.tight_layout()
plt.show()
```

3.4 Hybrid Models

3.4.1 LSTM + GRU

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, GRU, Dense, Concatenate, Flatten, Input
from tensorflow.keras.models import Model
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

```
# Load the air quality data
air_quality_data = pd.read_csv(r'C:\Users\tvams\OneDrive\Desktop\Imputation methods\Step-3.csv')
```

```
# Perform min-max normalization
scaler = MinMaxScaler()
normalized_data = scaler.fit_transform(air_quality_data.drop(columns=['Date', 'AQI']))
X_normalized = normalized_data
y = air_quality_data['AQI']
```

```
# Reshape data for LSTM/GRU input
X_train, X_test, y_train, y_test = train_test_split(X_normalized, y, test_size=0.2, random_state=42)
X_train = X_train.reshape(X_train.shape[0], 1, X_train.shape[1])
X_test = X_test.reshape(X_test.shape[0], 1, X_test.shape[1])
```

```
# Define LSTM and GRU layers
lstm_input = Input(shape=(X_train.shape[1], X_train.shape[2]))
lstm_output = LSTM(50, activation='relu', return_sequences=True)(lstm_input)
gru_input = Input(shape=(X_train.shape[1], X_train.shape[2]))
gru_output = GRU(50, activation='relu', return_sequences=True)(gru_input)
# Concatenate outputs of LSTM and GRU layers
concatenated = Concatenate(axis=1)([lstm_output, gru_output])
# Flatten concatenated output
flattened = Flatten()(concatenated)
# Dense layer for final prediction
output = Dense(1)(flattened)
# Define hybrid model
model = Model(inputs=[lstm_input, gru_input], outputs=output)
```

```
# Compile hybrid model
model.compile(optimizer='adam', loss='mse')
```

```
# Train hybrid model
history = model.fit([X_train, X_train], y_train, epochs=100, batch_size=32, validation_data=([X_test, X_test], y_test), verbose=1)
```

```
# Predict using hybrid model
y_pred = model.predict([X_test, X_test])

# Calculate evaluation metrics for hybrid model
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print("Hybrid Model Evaluation:")
print(f"MAE: {mae:.4f}")
print(f"MSE: {mse:.4f}")
print(f"RMSE: {rmse:.4f}")
print(f"R2 Score: {r2:.4f}")
```

```
# Plot training history
plt.figure(figsize=(10, 6))
plt.plot(history.history['loss'], label='Training Loss', color='blue')
plt.plot(history.history['val_loss'], label='Validation Loss', color='orange')
plt.title('Model Training History')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```

```
# Plot actual vs predicted AQI values
plt.figure(figsize=(10, 6))
plt.plot(y_test.values, label='Actual', color='blue')
plt.plot(y_pred, label='Predicted', color='red')
plt.xlabel('Index')
plt.ylabel('AQI')
plt.title('Actual vs Predicted AQI')
plt.legend()
plt.tight_layout()
```

```
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, color='red')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=2)
plt.xlabel('Actual AQI')
plt.ylabel('Predicted AQI')
plt.title('Actual vs Predicted AQI (LSTM+GRU)')
plt.tight_layout()
plt.show()
```

3.4.2 LSTM + GRU +RNN

```
[ ]: import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     from sklearn.preprocessing import MinMaxScaler
     from sklearn.model_selection import train_test_split
     from tensorflow.keras.models import Sequential, Model
     from tensorflow.keras.layers import LSTM, GRU, Dense, Concatenate, Flatten, Input, SimpleRNN
     from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
     import time
```

```
[ ]: # Load the air quality data
     air_quality_data = pd.read_csv(r'C:\Users\tvams\OneDrive\Desktop\Imputation methods\Step-3.csv')
```

```
[ ]: # Perform min-max normalization
     scaler = MinMaxScaler()
     normalized_data = scaler.fit_transform(air_quality_data.drop(columns=['Date', 'AQI']))
     X_normalized = normalized_data
     y = air_quality_data['AQI']
```

```
[ ]: # Reshape data for LSTM/GRU input
     X_train, X_test, y_train, y_test = train_test_split(X_normalized, y, test_size=0.2, random_state=42)
     X_train = X_train.reshape(X_train.shape[0], 1, X_train.shape[1])
     X_test = X_test.reshape(X_test.shape[0], 1, X_test.shape[1])
```

```
[ ]: # Define LSTM, GRU, and RNN layers
     lstm_input = Input(shape=(X_train.shape[1], X_train.shape[2]))
     lstm_output = LSTM(50, activation='relu', return_sequences=True)(lstm_input)

     gru_input = Input(shape=(X_train.shape[1], X_train.shape[2]))
     gru_output = GRU(50, activation='relu', return_sequences=True)(gru_input)

     rnn_input = Input(shape=(X_train.shape[1], X_train.shape[2]))
     rnn_output = SimpleRNN(50, activation='relu', return_sequences=True)(rnn_input)

     # Concatenate outputs of LSTM, GRU, and RNN layers
     concatenated = Concatenate(axis=1)([lstm_output, gru_output, rnn_output])

     # Flatten concatenated output
     flattened = Flatten()(concatenated)

     # Dense layer for final prediction
     output = Dense(1)(flattened)
```

```
[ ]: # Define hybrid model
     model = Model(inputs=[lstm_input, gru_input, rnn_input], outputs=output)
     start_time = time.time()
     # Compile hybrid model
     model.compile(optimizer='adam', loss='mse')
     end_time = time.time()
     compilation_time = end_time - start_time

     # Train hybrid model
     history = model.fit([X_train, X_train, X_train], y_train, epochs=100, batch_size=32, validation_data=([X_test, X_test, X_test], y_test), verbose=1)
     print(f"Compilation Time: {compilation_time:.2f} seconds")
```

```
[ ]: # Predict using hybrid model
     y_pred = model.predict([X_test, X_test, X_test])

     # Calculate evaluation metrics for hybrid model
     mae = mean_absolute_error(y_test, y_pred)
     mse = mean_squared_error(y_test, y_pred)
     rmse = np.sqrt(mse)
     r2 = r2_score(y_test, y_pred)

     print("Hybrid Model Evaluation:")
     print(f"MAE: {mae:.4f}")
     print(f"MSE: {mse:.4f}")
     print(f"RMSE: {rmse:.4f}")
     print(f"R2 Score: {r2:.4f}")
```

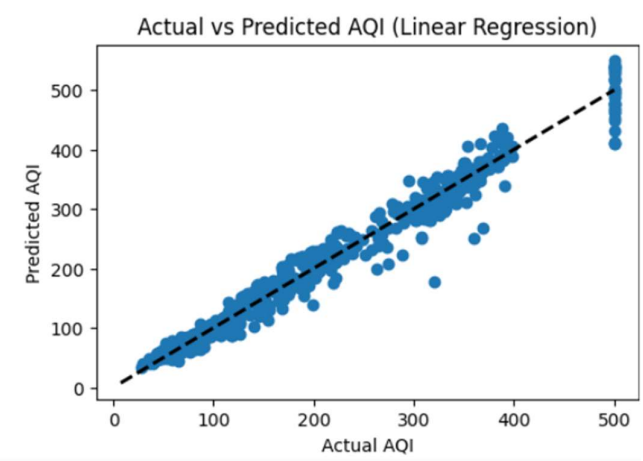
```
[ ]: # Plot training history
     plt.figure(figsize=(10, 6))
     plt.plot(history.history['loss'], label='Training Loss', color='blue')
     plt.plot(history.history['val_loss'], label='Validation Loss', color='orange')
     plt.title('Model Training History')
     plt.xlabel('Epochs')
     plt.ylabel('Loss')
     plt.legend()
     plt.show()
```

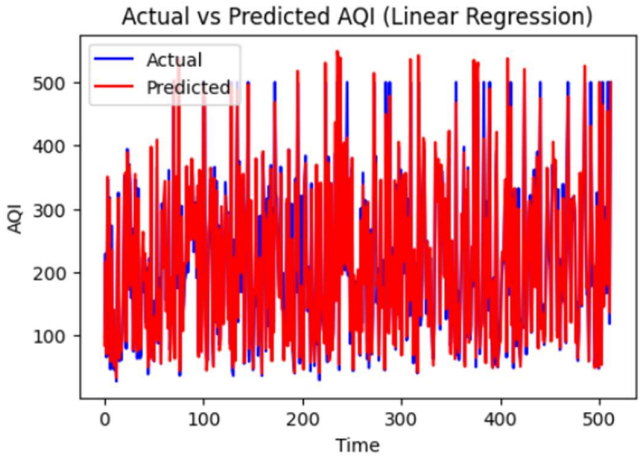
```
[ ]: # Plot actual vs predicted AQI values
     plt.figure(figsize=(10, 6))
     plt.plot(y_test.values, label='Actual', color='blue')
     plt.plot(y_pred, label='Predicted', color='red')
     plt.xlabel('Time')
     plt.ylabel('AQI')
     plt.title('Actual vs Predicted AQI(LSTM+GRU+RNN)')
     plt.legend()
     plt.tight_layout()
```

CHAPTER 4

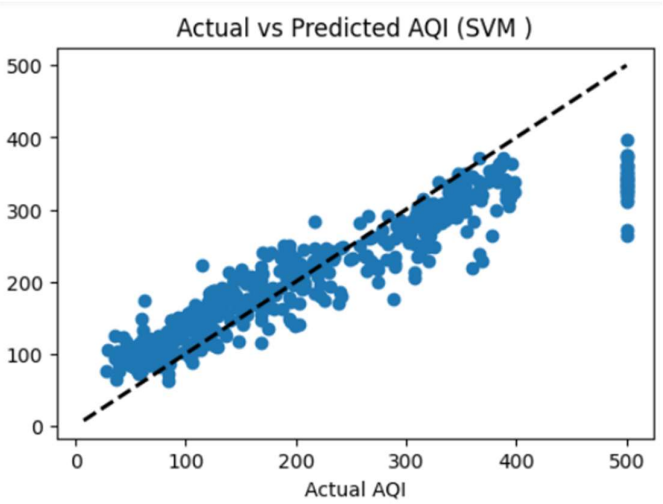
SNAPSHOTS

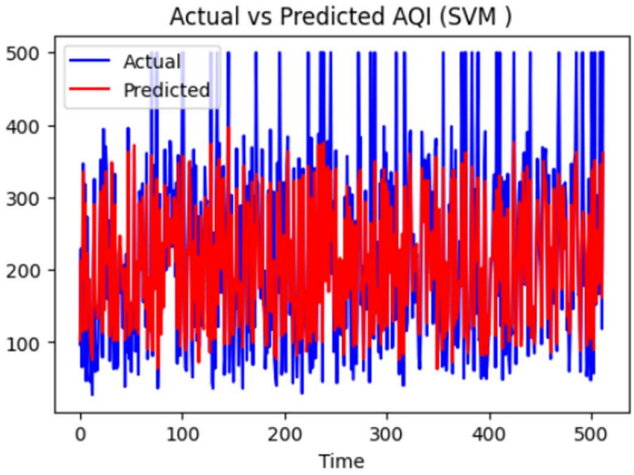
4.1 Linear Regression:



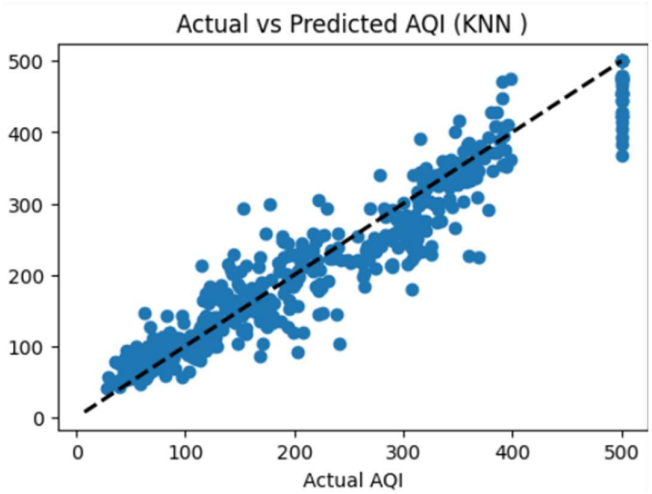


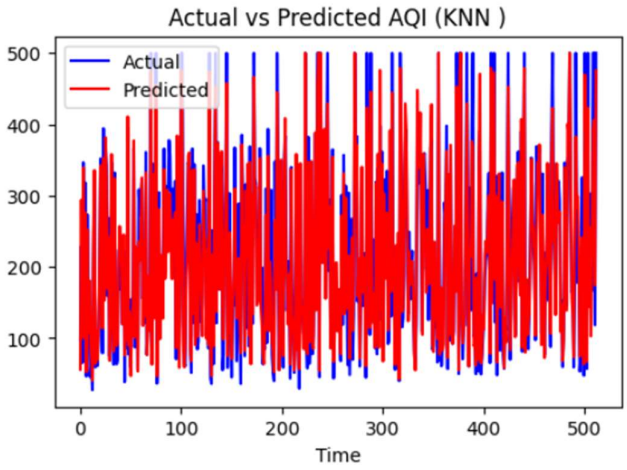
4.2 Support Vector Machine:



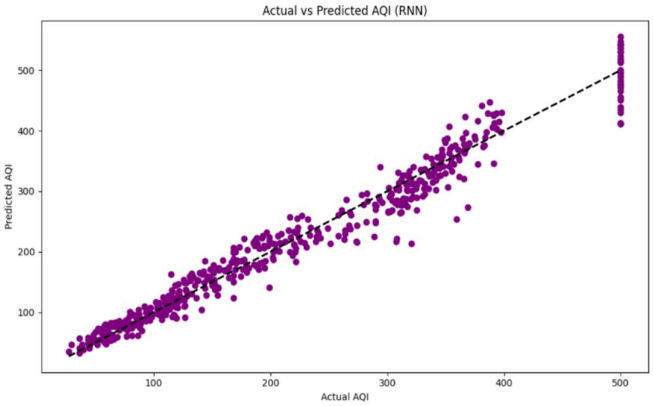


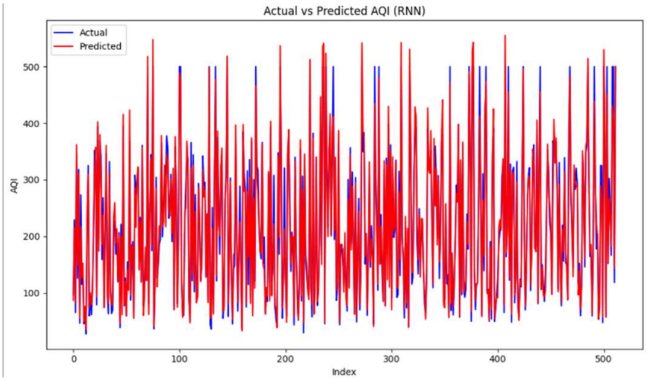
4.3 K-Nearest Neighbor:



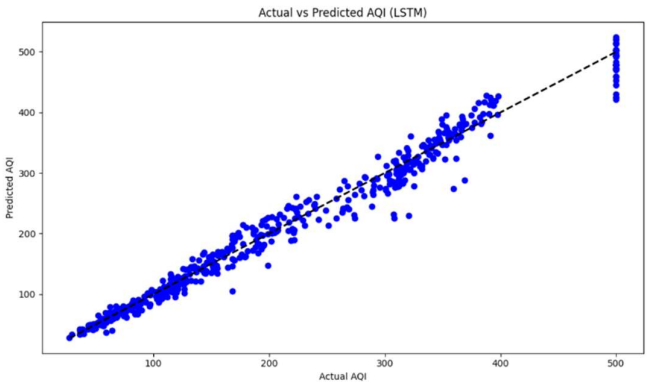


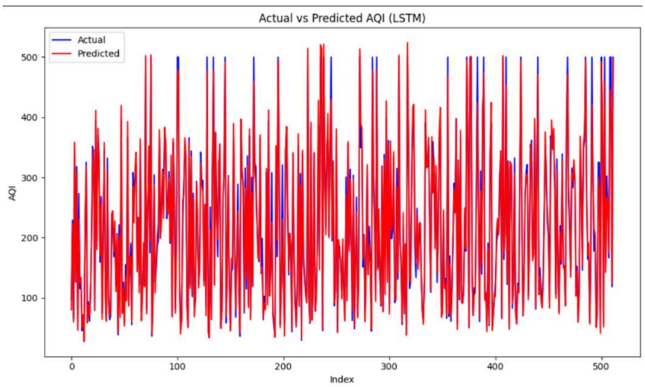
4.4 Recurrent Neural Network:



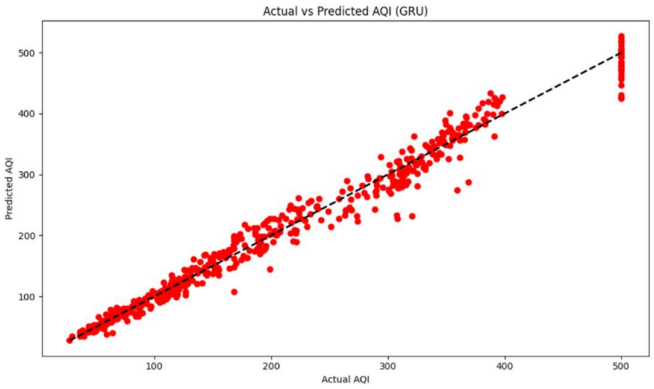


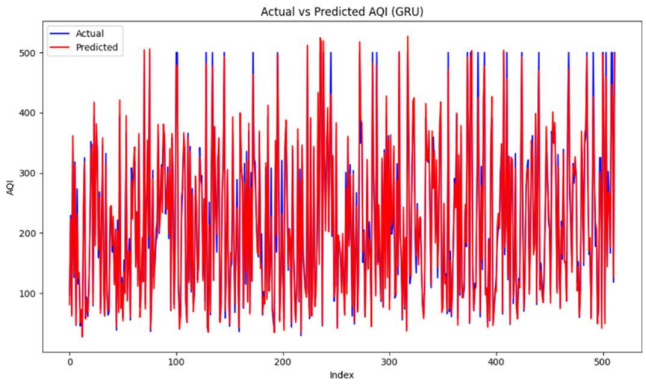
4.5 Long Short-Term Memory:



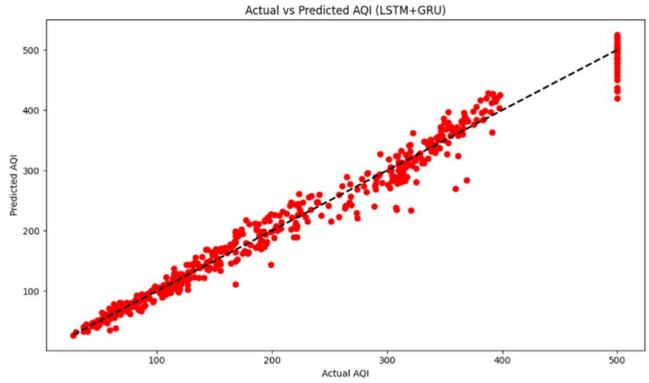


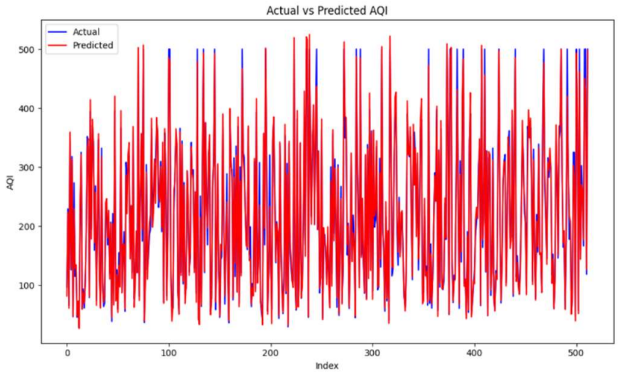
4.6 Gated Recurrent Units:



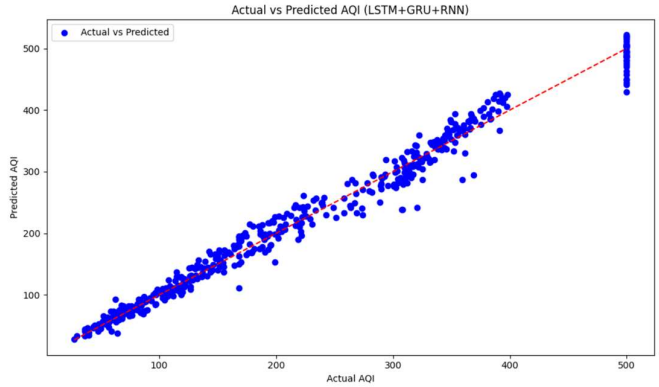


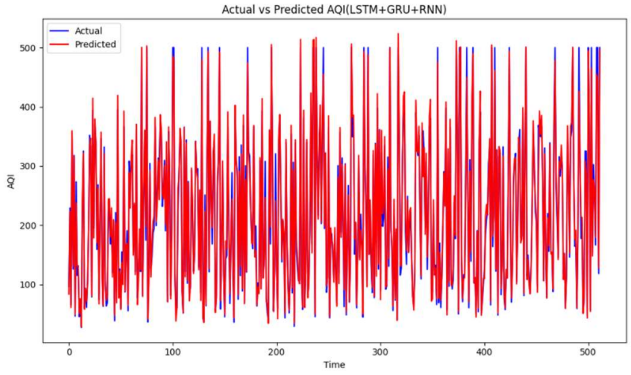
4.7 LSTM + GRU





4.7 LSTM + GRU + RNN





4.8 Evaluation Results

Models	Time(secs)	MSE	MAE	RMSE	R-Squared
Linear Regression.	0.1	639.95	16.87	25.25	0.9627
Support Vector Machine.	1.65	4777.19	50.04	68.95	0.7253
K-Nearest Neighbor.	0.2	1681.5	29.22	40.97	0.9025
Recurrent Neural Network(RNN).	0.11	514.91	16.4	22.69	0.9667
Long Short Term Memory(LSTM).	0.83	335.7	12.89	18.32	0.9702
Gated Recurrent Unit(GRU).	0.2	364.8	13.66	19.09	0.9788
LSTM + GRU.	0.09	341.52	13.06	18.48	0.9801
LSTM + GRU + RNN.	0.12	295.65	12.37	17.19	0.9828

Table 4. Evaluation Results

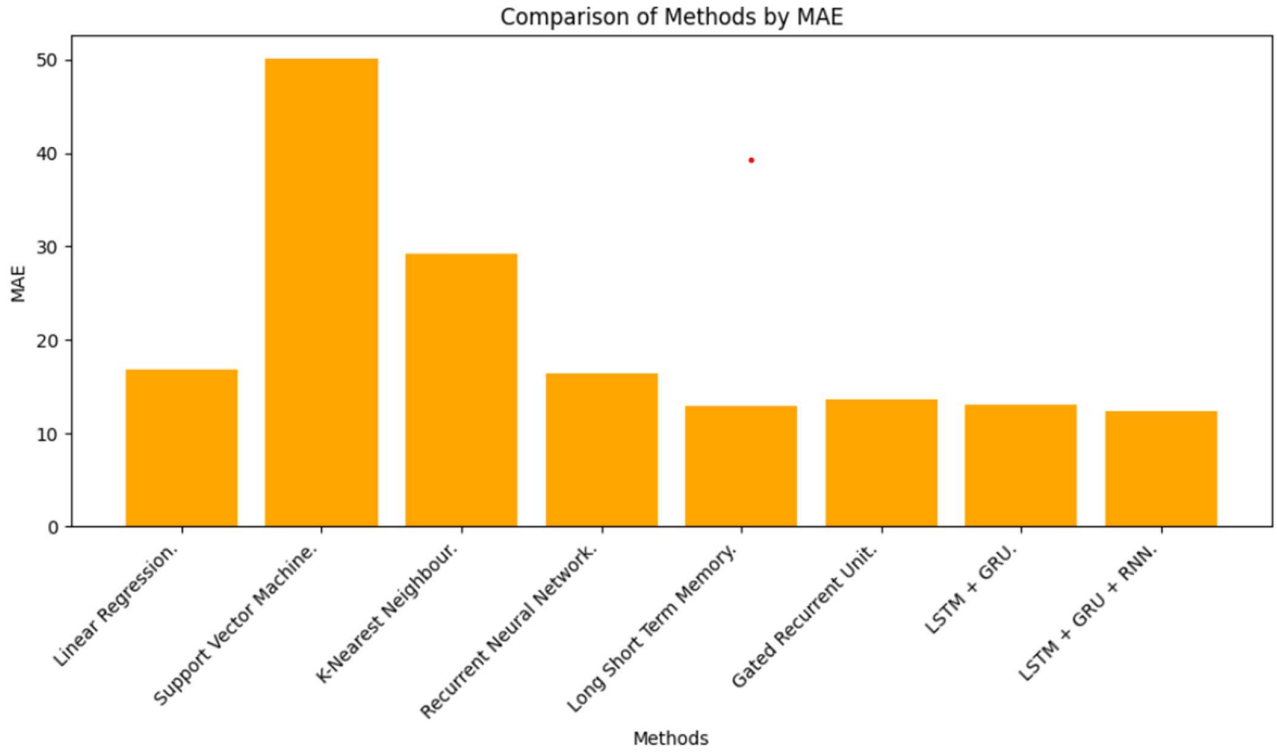


Fig 3: Comparison of MAE Values.

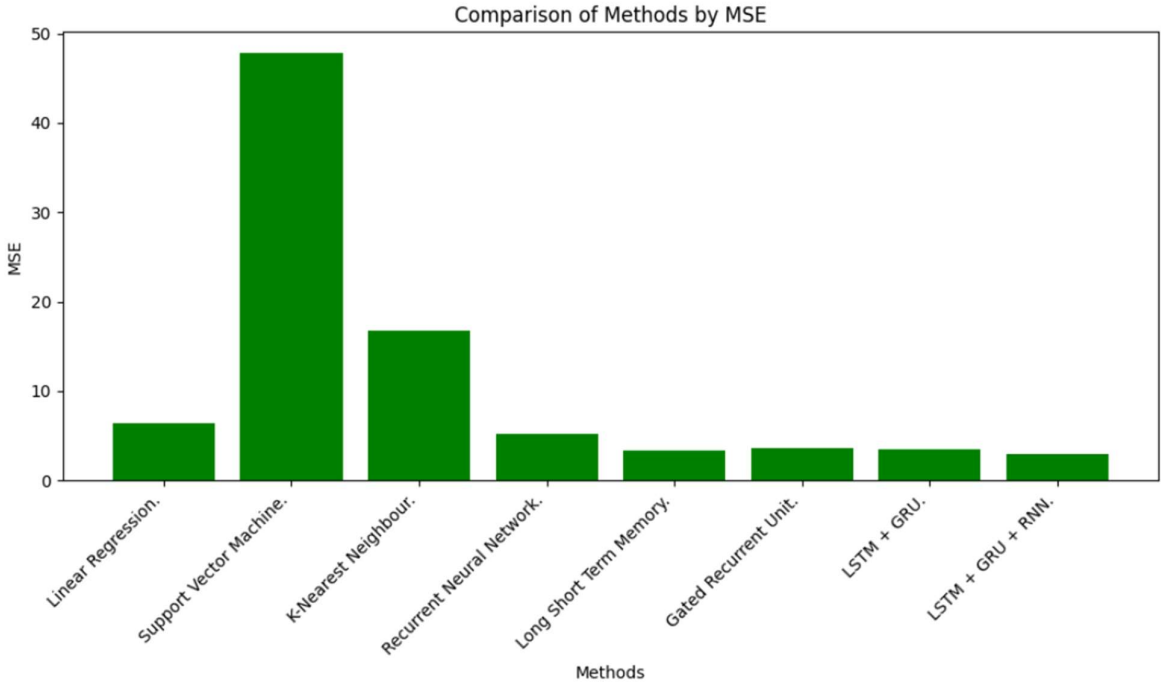


Fig 4: Comparison of MSE Values.

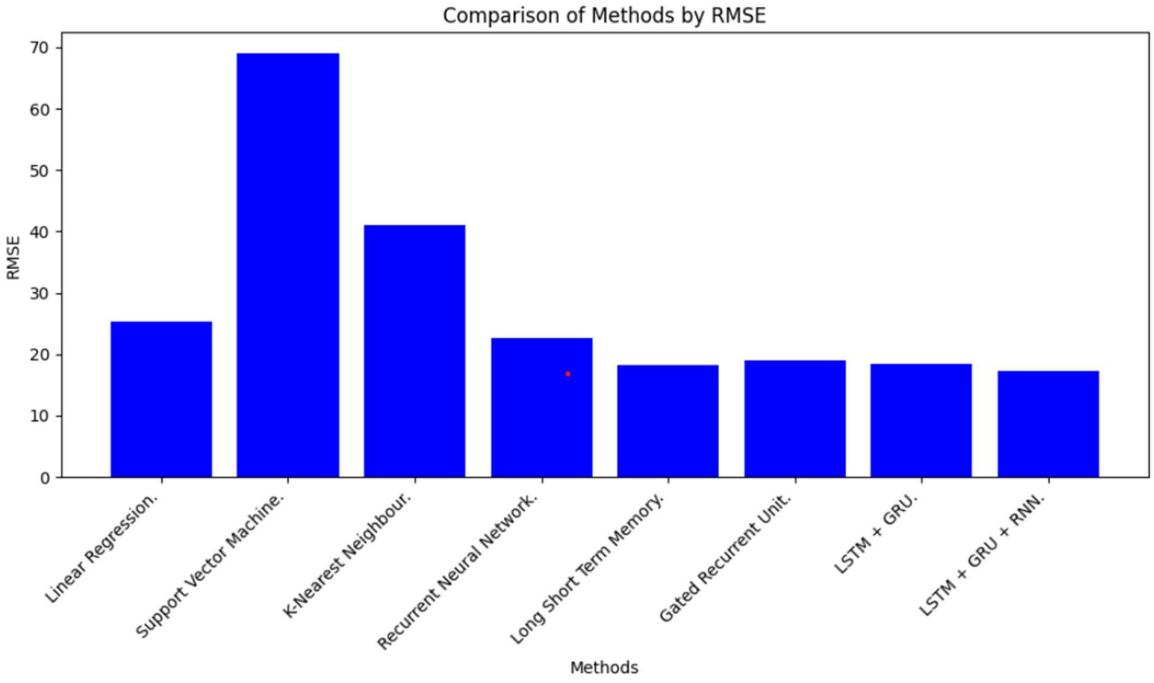


Fig 5: Comparison of RMSE Values.

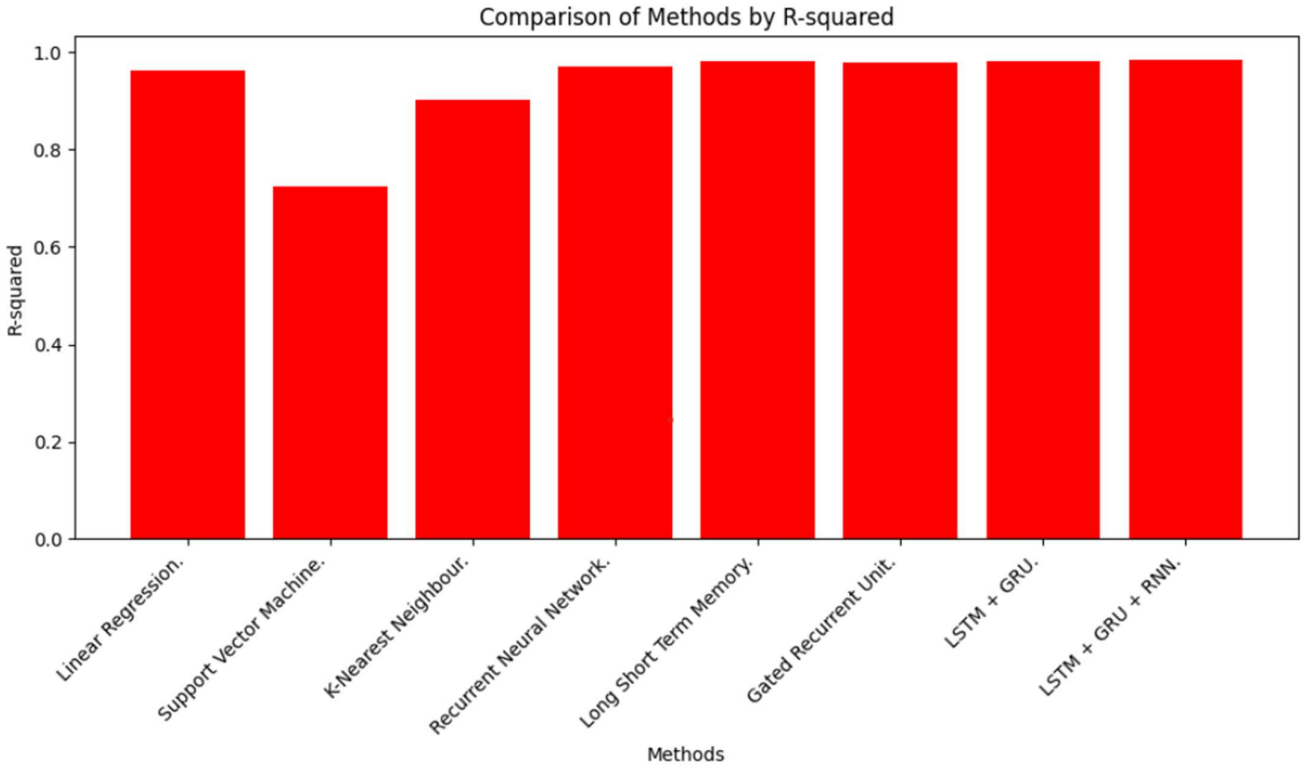


Fig 6: Comparison of R-Squared Values.

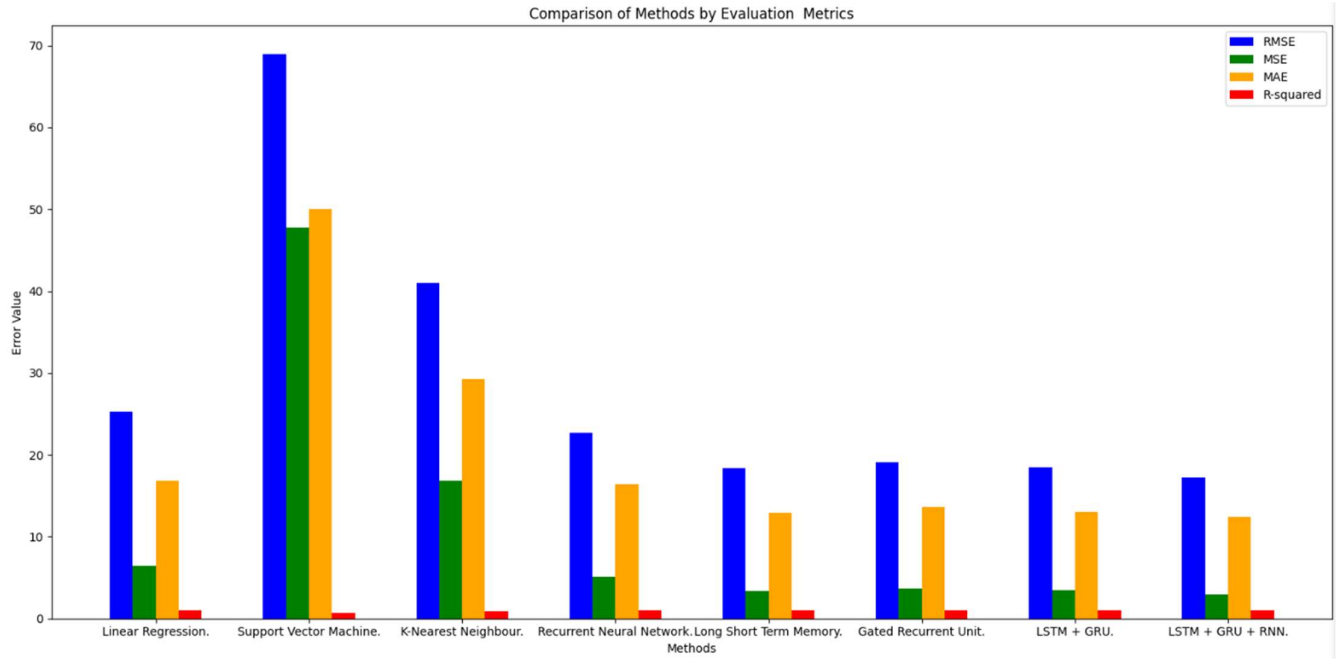


Fig 7: Comparison of Evaluation Metrics.

CHAPTER 5

CONCLUSION AND FUTURE PLANS

The conclusion of our project underscores the critical importance of anticipating the Air Quality Index (AQI) to safeguard public health. Despite the growing significance of this issue, the research landscape in the field remains inadequate. However, the study achieves a significant breakthrough by introducing a hybrid LSTM-GRU model, as well as an additional LSTM-GRU-RNN model, developed by our team, both of which exhibit impressive performance metrics.

Moving forward, our intentions involve actively developing the proposed approach beyond its current limitations. This expansion includes implementing the hybrid LSTM-GRU-RNN model, alongside the LSTM-GRU model, to predict AQI in a wide range of urban contexts beyond the initial study location. Furthermore, there is a strong emphasis on enhancing the reliability of future projections by advocating for the use of solid and reputable data sources.

Looking forward, the envisioned trajectory entails a proactive and expansive approach towards leveraging the hybrid model's capabilities. This includes extending its application to predict AQI in diverse urban landscapes, transcending the confines of the original study area. By encompassing varied geographical and environmental factors, the aim is to develop a versatile and adaptable framework capable of addressing the multifaceted challenges posed by AQI prediction.

Furthermore, paramount importance is attributed to the enhancement of data reliability and robustness in future predictions. This necessitates advocating for the utilization of high-quality, comprehensive datasets sourced from reputable sources. By anchoring predictions on reliable data foundations, the aim is to mitigate uncertainties and improve the overall accuracy and confidence in AQI forecasts.

In essence, the outlined future plans encapsulate a holistic approach towards advancing AQI prediction methodologies. By combining innovation, rigorous evaluation, and a commitment to data integrity, the research endeavours to not only bridge existing knowledge gaps but also pave the way for more informed and effective strategies in safeguarding public health against the detrimental effects of air pollution.

CHAPTER 6

REFERENCES

[1] Juxin Cao,Uzair Aslam Bhatti, Siling Feng, Mengxing Huang, Ahmad Hasnaing,"Air Quality Index Predictions with a Hybrid Forecasting Model: Combining Series Decomposition and Deep Learning Techniques" in IEEE 6th International Conference on Pattern Recognition and Artificial Intelligence (PRAI) , 2023 {keywords: AQI prediction, Air Quality Index, Hybrid Forecasting Model, LSTM, Deep Learning, Machine learning, Support Vector Machine}

[2] Yuan Zhongjie, Wang Shengwei, Wang Ze, "Air quality prediction method based on the CS-LSTM" in International Conference on Data Science and Information Technology, 2022 {Keywords: Component, air quality prediction, AQI, Prediction Accuracy, Optimal Parameters, LSTM model, CS-LSTM model}

[3] Guohui Li, Yuze Tang, Hong Yang, "A new hybrid prediction model of air quality index based on secondary decomposition and improved kernel extreme learning machine" in Chemosphere, 2022, {Keywords: Air quality index prediction, Multivariate multiscale dispersion entropy, Complete ensemble empirical mode decomposition with adaptive noise, Variational mode decomposition, Kernel extreme learning machine }

[4] Chenchen Li, Yu Bin Bao, "Research on Air Quality Prediction Based on Machine Learning" in 2nd International Conference on Intelligent Computing and Human-Computer Interaction (ICHCI), 2021, {Keywords: Machine learning, Random Forest, Air Quality Data, Air Quality Prediction, Air quality prediction, Regression algorithm}

[5] R Janarthanan, P Partheeban, K Somasundaram, P Navin Elamparithi, "A deep learning approach for prediction of air quality index in a metropolitan city" in Sustainable Cities and Society, 2021, {Keywords: Air quality index, Deep learning, LSTM model, National Air Monitoring Program}

[6] Ruijun Yang, Xueqi Hu, Lijun He, Shang Hai, “Prediction of Shanghai air quality index based on BP neural network optimized by genetic algorithm” in International Symposium on Computational Intelligence and Design, 2020, {Keywords: bp neural network, AQI (air quality index), Mean Square Error, Air QualityIndex}

[7] Wang Zhenghua, Tian Zhihui, “Prediction of air quality index based on improved neural network.” in International Conference on Computer Systems, Electronics and Control (ICCSEC), 2017, {Keywords: Xuchang city, air quality index prediction, Genetic algorithm, neural network}

[8] Chuanting Zhang, Dongfeng Yuan, “Fast Fine-Grained Air Quality Index Level Prediction Using Random Forest Algorithm on Cluster Computing of Spark.” in IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing, 2015, {Keywords: Big data, air quality prediction, Spark, random forest}

[9] Dataset for Air Quality data: National Air Quality Index (cpcb.gov.in)

[10] Dataset for meteorological data: POWER | Data Access Viewer (nasa.gov)

[11] Information about AQI: Air Quality Index - Wikipedia

CHAPTER 7

APPENDIX – BASE PAPER

Environmental Pollution 315 (2022) 120404

Contents lists available at ScienceDirect

Environmental Pollution

journal homepage: www.elsevier.com/locate/envpol

Air Quality Index prediction using an effective hybrid deep learning model☆

Nairita Sarkar, Rajan Gupta, Pankaj Kumar Keserwani *, Mahesh Chandra Govil

Computer Science and Engineering Department, National Institute of Technology Sikkim, South Sikkim, Ravangla, Sikkim, India

ARTICLE INFO

Keywords:
Air quality index
Long short term memory
Gated recurrent unit
Linear regression
K-nearest neighbor
Support vector machine

ABSTRACT

Environmentalism has become an intrinsic part of everyday life. One of the greatest challenge to the environment's long-term existence is the air pollution. Delhi, the capital of India, has experienced decreasing of air quality for several years. The poor air quality has a significant impact on the lives of individuals. Air Quality Index (AQI) prediction can help to its beneficiaries in taking safeguards about their health before moving to any polluted area. In this study, a variety of data forecasting approaches is evaluated to predict the AQI value for Particulate Matter ($PM_{2.5}$) μm at a particular area of Delhi and several error-prone strategies such as R-Squared (R^2), Mean Absolute Error (MAE), and Root Mean Square Error (RMSE) methods are catalogued. In the proposed approach two deep learning models like Long-Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) are combined to predict the AQI of the environment. Several stand alone machine learning (ML) and deep learning (DL) models such as LSTM, Linear-Regression (LR), GRU, K-Nearest Neighbor (KNN) and Support Vector Machine (SVM) are also trained on the same dataset to compare their performances with the proposed hybrid (LSTM-GRU) model and it is found that the proposed hybrid model shows supremacy in the performance with the MAE value 36.11 and R^2 value 0.84.

1. Introduction

Air pollution is a significant environmental issue in different parts of the world. When dangerous or excessive amounts of substances, such as gases, particles, and biological molecules, are emitted into Earth's atmosphere, air pollution occurs. Nitrogen Dioxide (NO_2), Carbon Monoxide (CO), Carbon Dioxide (CO_2), Ozone (O_3), Sulfur Dioxide (SO_2) are the fine Particulate Matters as well as pollutants that causes air pollution (Zhu et al., 2017). On the other hand, air pollution, as much as any other health hazard, is responsible for a large proportion of early deaths in India (Glencross et al., 2020), (Pozzer et al., 2020). Growth in population density, expansion of industry and motor vehicles and increase in annual average temperature are the foremost influencing factors of air pollution. Additionally, the process of urbanisation has a detrimental effect on air quality.

Because of the harmful effects that pollution has on human health, the public is highly concerned about the future trend of air quality (Zou et al., 2019). Air pollution causes several diseases such as asthma, weakens lung function, increases cardiopulmonary illnesses (Yuan and Liu, 2014) and also escalates mortality rates. Numerous recommendations are there to shield the population from severe pollution, for example: i) public participation in outdoor activities should be reduced, ii) patients with severe diseases like heart disease, respiratory disease, pregnant women, children should stay in home more times. It is necessary to anticipate air pollution for both warning about and reducing pollution in people's daily lives.

The Air Quality Index, often known as the AQI, is a statistical measurement used to determine the quality of breathing air in our surroundings that consolidates the concentrations of various contaminants into a single numerical form (Zhu et al., 2017). It is used to investigate the long-term health effects of air pollution on a person's health as it explores the relationship among human health and air quality. The new ambient air quality standards (GB3095-2012), which covers six pollutants including Ozone (O_3), Carbon Monoxide (CO), Nitrogen Dioxide (NO_2), Sulfur Dioxide (SO_2), and $PM_{2.5}$, PM_{10} (particulate matter with a diameter less than or equal to 10 μm), are used to calculate the AQI (Zhu et al., 2017). The most commonly used air quality evaluation indexes are criteria air pollutants and comprehensive indices (Zhu et al., 2018).

☆ This paper has been recommended for acceptance by Pavlos Kassomenos.

* Corresponding author.
E-mail addresses: phcs220002@nitsikkim.ac.in (N. Sarkar), b180030@nitsikkim.ac.in (R. Gupta), pankajkeserwani.cse@nitsikkim.ac.in (P.K. Keserwani), director@nitsikkim.ac.in (M.C. Govil).

https://doi.org/10.1016/j.envpol.2022.120404
Received 30 June 2022; Received in revised form 27 September 2022; Accepted 6 October 2022
Available online 11 October 2022

There has been a considerable deterioration in the quality of air in several cities of India throughout the years (Singh and Chauhan, 2020). It is very much critical to monitor overall air quality within the region and provide warnings when necessary. Whenever it comes to interpreting air pollution, raw data is difficult to comprehend. To address this issue, a quantitative measurement - AQI is being developed by the research community.

AQI prediction methods for air pollutants may be classified into three categories: i) ML based methods, ii) DL based methods and iii) Hybrid methods.

ML based method is a type of computer program that can identify patterns based on past data. A multi-source ML method was implemented by D. Iskandaryan et al. (2020) to evaluate the AQI scores of large metro polish neighborhoods. They used three significant sample sets (MNR-Air-HCM, SEPHLA-Medieval 2019 and MNR-HCM) to evaluate random forest performance. The researchers discovered that air quality stability has a powerful link between increasing air purity and may also assist to stabilise the environment. Van et al. (2022) employed a strategy for handling data on air pollution and increasing the AQI forecasting process using various ML based models like: Decision-Tree (DT), Random Forest (RF), and Extreme Gradient Boost (XGBoost) and the results are analysed through MAE, RMSE, and R^2 methods. XGBoost beated in foreseeing the AQI values than the other algorithms with RMSE = 0.03684. Mahalingam et al. (2019) discussed the difficulty of forecasting AQI so that pollution can be controlled before it worsens. To resolve it they utilized two ML Algorithms: Neural Network (NN) and Support Vector Machine (SVM) on the data taken from Central Pollution Control Board (CPCB). Liu et al. (2019) used Support Vector Regressor (SVR) and Random Forest Regressor (RFR) to predict Beijing's AQI and the concentration of Nitrogen Dioxides (NO_2). The regression models' performance was evaluated using RMSE and R^2 and SVR-based model predicted the value of AQI better than the RFR-based model with R^2 = 0.9766 and RMSE = 7.666. Pant et al. (2022) made an accurate prediction of the air quality in Dehradun, Uttarakhand by employing supervised machine-learning methods. The test results demonstrated that the decision-tree had a higher degree of accuracy, with a precision of 98.63%. It was interesting to see that the logistic regression had the highest precision of 91.78% for the expectation of air quality. Whereas, Castelli et al. (2020) examined the contaminant and particle levels and estimated the air quality score using Support Vector Regression (SVR) and Radial-basis-function (RBF) and it was proved that SVR was giving the most accurate and trustworthy predictions with accuracy rate was 94.1%. Gocheva-Iliev et al. (Gocheva-Ilieva et al., 2014) suggested the prediction of AQI using the Linear-Regression Algorithm and Gradient-Boost Algorithm. The Naive Forecast method was also used, but the Gradient Boost Algorithm was more accurate with 96% accuracy. This project helped to find the main pollutant that is causing pollution. Campbell-Lendrum et al. (Campbell-Lendrum and Prüss-Ustün, 2019) presented the use ML to predict $PM_{2.5}$ μm level from weather data in a non metropolitan city at a high elevation (Quito, Ecuador). Auto-regression was used to forecast $PM_{2.5}$ μm levels based on past $PM_{2.5}$ μm levels. SVM algorithms, Decision Tree (DT) were able to predict $PM_{2.5}$ μm with accuracy score of about 89%. Bhalgat et al. (2019) presented an overview of many investigations done on air quality prediction using SVM, Decision Tree (DT), and Effective Algorithms on a static dataset. Due to use of static dataset, there was no specific parameter on which the quality of air is dependent that may cause differ in forecast in real-world settings. To solve this challenge, a web scraper might be used to retrieve real-time datasets for training models, enhancing accuracy and offering better results. To determine how well AQI prediction works, Nigam et al. (2015), came up with a method that combined protocols for processing air pollution data and effective machine learning methods. To find the best model for AQI expectations, the MAE, RMSE, and R^2 measurements were used to compare three computation models: DT, RF, and XGBoost. The suggested methods were tested using two different public datasets that were collected from different parts of India. XGBoost method gave better result than others to predict the AQI. So, XGBoost was chosen again for a low-cost device to estimate the AQI at fixed-site assessment units. Similarly, Janarthanan et al. (2021) used two ML algorithms Support Vector Machines and Neural Networks, to predict the AQI.

The previous paragraph explains the detail of ML based approaches used by the research community to predict the AQI value, this paragraph represents a brief review on DL based methods utilized for AQI prediction. Due to its supremacy in terms of accuracy while learn with massive amounts of data, deep learning is becoming increasingly popular. In Sigamani and Venkatesan (2022), Sigaman et al. demonstrated the usage of features like $PM_{2.5}$ μm, PM_{10} μm, and other gaseous components used to predict AQI. The Auto-Regressive Integrated Moving Average (ARIMA) model, Auto Regression model, and Linear Regression model were employed as deep learning algorithms. On the other hand, Zhou et al. (2019) build an AQI prediction model based on Convolutional Neural Network (CNN), an attention mechanism, and GRU in their work. CNN was used to extract features from the input data and Gated Recurrent Unit (GRU) was used to predict the AQI. Weather and air quality data from January 1, 2017 at 00:00 to September 30, 2020 at 23:00 in the Chinese city of Shijiazhuang were collected and were applied to GRU prediction model to prove the better accuracy and it was compared with the performance of other models and the results of the proposed model showed the best overall performance with MAE = 6.099281, MSE = 90.781522, EVS = 0.972560, R^2 = 0.972495.

The hybrid methods mainly includes the combination of more than one ML or DL models. Chen et al. (Chen et al., Hong) used hybrid model to estimate the contamination, climate, and compounds was collected from the Weather Research and Forecasting-Chem (WRF-Chem) model. The partial mutual information (PMI) based input variable selection (IVS) hybrid model, also known as PMI-IVS model has the superior outcome than the current models from a technical standpoint with Calibration: 0.56, Validation: 0.52. On the other hand, Alireza et al. (2021) makes a use of Hybrid Single Degradation (HSD) and Hybridization Two Phase Decomposition (HTPD) model to make a prediction about the air quality record of Orumiyeh, Turkey, one day in advance. According to the results, the Complete Ensemble Empirical Mode Decomposition with Adaptive Noise-Environmental Liabilities Management (CEEMDAN-ELM) model and the Complete Ensemble Empirical Mode Decomposition with Adaptive Noise-General Regression Neural Network (CEEMDAN-GRNN) model both were successful in accurately forecasting AQI series information by employing HSD models. The HTPD shows the supremacy in accuracy and the value of R^2 = 0.98, RMSE = 4.13 and MAE = 2.99 in the training phase and the value of R^2 = 0.74, RMSE = 5.45 and MAE = 3.87 in the testing phase. An RNN-LSTM model proposed by Partheeban et al. (Partheeban) that can predict the amount of pollution in the air at any given hour and applied in a certain part of Delhi to figure out the AQI. The LSTM model got its information from time sequences of four weather parameters and the pollution levels. Similarly, Wu et al. (Wu and Lin, 2019) proposed a novel optimal-hybrid model in their study for AQI forecasting based on sample entropy (SE), secondary decomposition (SD), least squares support vector machine which is optimized by Bat-algorithm (BA-LSSVM) and LSTM neural network. The proposed SD-SE-LSTM-BA-LSSVM model achieved the supremacy in accuracy while comparing with the other hybrid models. The calculated error rates for the proposed model are: MAE = 6.6885, RMSE = 8.8920, MAPE = 0.0877 for the city Beijing and calculated MAE, RMSE, MAPE values for Guilin are 3.8036, 4.4396, 0.0880 respectively. For more improvement of the accuracy of AQI G Li et al. (2022) introduced an innovative hybrid AQI prediction model named CEEMDAN-mvMDE-BVMD-RSO-KELM by employing complete ensemble empirical mode decomposition with adaptive noise (CEEMDAN) for AQI decomposition, multivariate multiscale dispersion entropy (mvMDE) for calculating the complexity of every decomposed element, variational mode decomposition (VMD) optimized by Bald Eagle Search (BES) algorithm i.e., BVMD for selecting the decomposition

level and penalty factor and kernel extreme learning machine optimized by Rat Swarm Optimizer (RSO) algorithm (RSO-KELM) for anticipating the intrinsic mode function (IMF) components which are obtained from the AQI decomposition.

Table 1 summarises some of the mentioned works performed and evaluated to predict the Air Quality Index (AQI).

It is evident from above literature survey that a limited research are expressed for AQI prediction on various datasets. Now it is becoming very difficult to monitor and to predict the AQI, especially for urban areas due to industrial developments and increasing transportation. In order to predict AQI in the urban areas this work first uses samples gathered from the Central Pollution Control Board (CPCB). After that a hybrid model called hybrid LSTM-GRU model is being developed which uses R2, MAE, and RMSE approaches to assess the performance of the developed model, and finally compares the performance of the proposed integrated LSTM-GRU model with some developed ML and DL based models as well as other recent existing approaches. In the proposed approach a step-by-step workflow is being presented for the transformation of raw data into an integrated hybrid feature space to enhance the prediction of Air Quality Index of a certain region of Delhi with minimal error. For conducting the experiments raw data are taken from the Central Pollution Control Board's official website and the several pre-processing and integration processes such as data imputation, data aggregation, and data normalization are being conducted. Hereafter, rich features are selected as a next from the pre-processed dataset by discarding features with less feature importance score. The selected feature vector is fed as the input to the input layer of the ML and DL models. For modifying these time series data, numerous ML and DL models such as: linear regression (LR), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Gated Recurrent Unit (GRU) and Long Short-Term Memory (LSTM) networks are deployed and further standalone GRU and LSTM models are combined together to get better result for AQI forecasting. The proposed hybrid LSTM-GRU model shows the supremacy in performance with minimal error rate. The significant contributions of this work are provided below:

Table 1
Summarizing of calibrated works to predict the AQI.

Ref.	Method	YOP	Result(s)
Iskandaryan et al. (2020)	SVM, RF, EGB, LightGBM and CatBoost	2020	RMSE = 9.3953
(Chen et al., Hong)	PMI-IVS and PEK-based ML	2021	Calibration: 0.56, Validation: 0.52
Li et al. (2019)	ML algorithms	2019	AR-Kalman model: MAE = 2.87, MAPE = 3.18, RMSE = 3.25, MSE = IMM Model: MAE = 0.54, MAPE = 0.55, RMSE = 0.95, MSE = 5.23
Van et al. (2022)	DT, RF, and GB	2022	RMSE = 0.03684
Mahalingam et al. (2019)	ANN, SVM	2019	RMSE = 17.21,RMSE = 17.88, RMSE = 13.91
Liu et al. (2019)	SVR, RFR	2019	SVR r = 0.9887, R^2 = 0.9766 RMSE = 7.666, RFR r = 0.9823, R2 = 0.9633, RMSE = 9.602
Srivastava et al. (2018)	LR, SDG, RF, DT, SVM, CNN, GB,	2018	R^2 = 0.65646, 0.65922, 0.67, 0.62, 0.69275, 0.68478, 0.69647, 0.69275
Pant et al. (2022)	DT, LR	2022	DT = 98.63% LR = 91.78%
Castelli et al. (2020)	SVR-RBF, PCA, SVR-RBF	2020	94.1%
Kleine Deters et al. (2017)	Multiple Linear Regressive model	2017	MSE = 15.0
Alireza et al. (2021)	HSD, HTPD, CEEMDAN-ELM, CEEMDAN-GRNN	2020	R^2 = 0.74 RMSE = 5.45 MAE = 3.87
Londhe (2021)	Multiple regression, RMSE, KNN	2021	RMSE = 52.34, R^2 = 0.89, MAE = 24.79
Gocheva-Ilieva et al. (2014)	LR and GB	2014	Accuracy = 96%
Campbell-Lendrum and Prüss-Ustün (2019)	SVM and DT	2019	Accuracy = 89%
Bhalgat et al. (2019)	SVM, DT	2019	MSE = 166.358
Sigamani and Venkatesan (2022)	MLR, ANN, SVM	2022	R^2 = 0.923, 0.931, 0.953
Liu et al. (2018)	RF, SVM,DT, GB,	2018	RMSE = 9.8
Janarthanan et al. (2021)	SVM,NN	2021	RMSE = 10.995, R^2 = 0.570
Zhou et al. (2019)	CNN, GRU	2021	MAE = 6.099281, MSE = 90.781522, EVS = 0.972560, R^2 = 0.972495
(Partheeban)	RNN-LSTM model	2021	MAE = 1 R^2 = 0.89

- We have accumulated the air quality data from a particular city in India and built the hybrid Long-Short-Term Memory-Gated Recurrent Unit (LSTM-GRU) model to predict the AQI value by taking into account the pollutant content in the air as well as many meteorological characteristics, e.g., temperature, wind speed, and relative humidity, amongst others.
- Permutation feature importance method is adopted to find out the feature importance score of each feature of the dataset and the feature with least importance score is discarded.
- Various standalone ML and DL models: LR, KNN, SVM, LSTM, GRU as well as the Hybrid LSTM-GRU model are utilized to receive the chosen feature vector as input and the effectiveness of all the models has been evaluated.
- Efficiency of each of the model is analysed on the basis of RMSE, MAE and R^2 values and the proposed hybrid model shows the best performance w.r.t the MAE and R^2 values.

1.1. Organization of the paper

The remaining parts of the paper are divided into the following sections: materials and methods associated with this study are outlined in Section 2. Section 3 shows the results and the implementation planning. The conclusive comparison of performance is presented in Section 4. The paper is brought to an end in conclusion presented in Section 5, which also discusses the future work direction.

2. Materials and methods

The step by step process flow for developing an AQI prediction model is presented in this section. All involved steps or modules required for implementing the proposed model are exhibited in Fig. 1. The flow diagram consists of several modules: reading of AQI data, data preprocessing, dataset splitting, then utilize the ML and DL models to predict AQI and finally choose the best model on the basis best performance. All the associated steps are explained subsequently.

2.1. Data collection

The process of collecting and evaluating information is referred to as data collection. In this study two sets of data are used. Data is collected from official portal of the Indian government, the Central Pollution Control Board. The dataset includes pollutant concentration and climate data collected every hour for seven years (from March 2015 to December 2021). The dataset contains 2482 number of samples of weather data from Delhi taken every day basis. In the dataset, there are 14 climatic factors, like temperature, wind-speed, humidity, and so on. The Central Pollution Control Board, where data for 18 states, 102 cities,

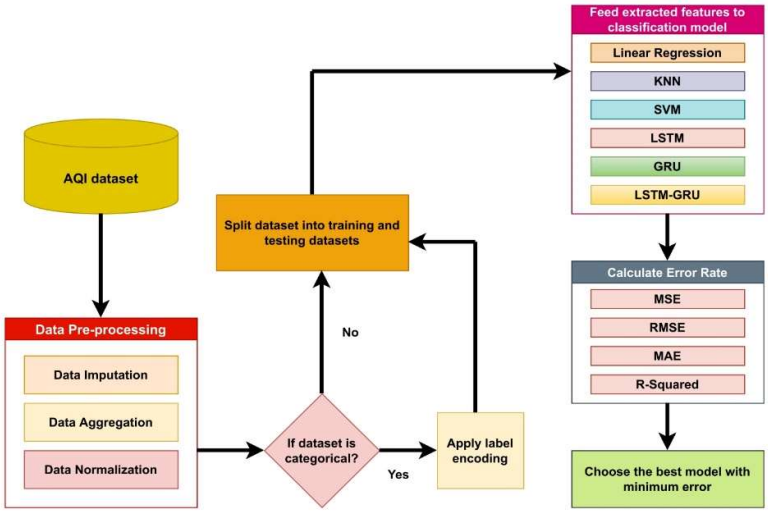


Fig. 1. Proposed process flow of developing an AQI prediction model.

and 170 stations is publicly accessible, is consulted in order to collect the information used to determine the concentration of pollutants and the meteorological parameters. The station considered for data collection is Netaji Subhas University of Technology (NSUT) which was formerly known as Netaji Subhas Institute of Technology (NSIT), situated in Sector 3 of Dwarka. In comparison to the data from other stations, those data from NSUT are quite dense, with some non-assigned values, totalling anywhere from a few days to a few weeks on an annual basis. As a result, data spanning seven years is used in order to accomplish the goal. The chronology for the data begins on March 23, 2015 and continues through December 31, 2021. The number of data points were collected hourly then converted to daily by taking the hourly average value.

Table 2 depicts a list of considered air pollutants: Carbon Monoxide (CO), Nitrogen Oxide (NO), Nitrogen dioxide (NO_2), Nitrogen Oxides (NO_x), Sulfur dioxide (SO_2) and Ozone (O_3). CO an uncolored, unperfumed and tasteless gas which is generated by burning fuels. It is extremely poisonous and a bit less dense than air. CO is measured by the unit mg/m^3 which tells the amount of CO in milligram per unit cubic meter. NO, NO_2 and NO_x belongs to the family of extremely reactive, toxic gases which are also produced by flickering fuels in high temperature. The quantity of these gases in air is measured by the unit microgram per cubic meter of air i.e. $\mu g/m^3$. Similarly, SO_2 is also a poisonous gas that gives off the odour of burnt matches and measured by $\mu g/m^3$. O_3 is a highly reactive gas comprised of three oxygen atoms and belongs to both the natural and artificial substances. It has many bad impacts on human health like chest pain, coughing, lung infections etc. O_3 is measured by $\mu g/m^3$.

Table 3 presents meteorological parameters: temperature, wind speed, relative humidity and solar radiation which have a great impact on air pollution. The brief idea about each of the meteorological parameter is enlisted below:

Table 2
List of pollutants considered.

Sl. No.	Parameters	Units
1	NO (Nitrogen Oxide)	$\mu g/m^3$
2	NO_2(Nitrogen dioxide)	$\mu g/m^3$
3	NO_x(Nitrogen Oxides)	$\mu g/m^3$
4	SO_2(Sulfur dioxide)	$\mu g/m^3$
5	CO(Carbon Monoxide)	mg/m^3
6	O_3(Ozone)	$\mu g/m^3$
7	$PM_{2.5}$(Particulate Matter 2.5 μm)	$\mu g/m^3$

Table 3
Meteorological parameters.

Sl. No.	Parameters	Units
1	Temperature	°C
2	WS (Wind Speed)	m/s
3	RH (Relative Humidity)	%
4	SR (Solar Radiation)	W/m^2

- **Temperature:** A physical quantity that indicates climatic conditions, whether it is cold or hot, is referred to as temperature. High temperature influences the motion of air which makes some consequences on air pollution. It is measured by degree Celsius (°C) or degree Fahrenheit (°F).
- **Wind Speed:** In meteorology, wind speed is a fundamental atmospheric parameter brought on by air shifting from a state of high to low pressure, usually happens as a result of temperature variations. Wind speed is basically a rate in which air is passing through an area. It is measured by meter per second (m/s).
- **Humidity:** The amount of water or moisture that is present in the air as water vapour is referred to as humidity. The quantity of hazardous or dangerous substances in the air increases with high humidity. Humidity is measured in percentage (%).
- **Solar Radiation:** Solar radiation is the electromagnetic radiation released by sun. This radiation makes some effects in temperature which in turns is related with air pollution. Solar radiation is measures by Watt per square meter (W/m^2).

The detail information of statistical measurement i.e. the count, mean, standard deviation etc. of the used dataset is illustrated in Table 4. "Count" calculates the total number of non-missing values for each corresponding feature. "Mean" is the average of all the observations used. The term "standard deviation" refers to a measurement of the dispersion of data from the mean.

2.2. Data pre-processing

A dataset contain important information as well as noises, outliers, and null values. These noises, outliers, and null values are affecting the

Table 4
Information of pollutants and meteorological parameters.

Index	NO	NO_2	NO_x	SO_2	CO	O_3	Temp	RH	WS	SR	$PM_{2.5}$
Count	2337.0	2338.0	2341.0	2365.0	2271.0	2346.0	2370.0	2267.0	2221.0	2370.0	2311.0
Mean	18.61	30.23	30.62	10.52	124.21	32.80	25.97	57.08	0.91	115.06	107.85
Std	20.54	14.48	21.05	7.28	253.52	20.12	7.25	18.17	0.42	76.44	73.13
Min	0.21	0.0	0.0	0.3	0.0	0.8	5.36	8.79	0.05	1.5	8.29
25%	6.49	20.02	16.92	5.88	0.59	20.14	20.312	43.95	0.63	69.65	56.875
50%	10.93	27.52	24.1	8.73	0.91	28.805	27.79	58.7	0.86	101.84	92.08
75%	21.91	37.635	37.96	12.82	2.29	40.67	31.375	70.74	1.14	139.63	137.03
Max	170.37	148.58	171.52	80.05	992.44	240.85	60.57	101.79	3.16	790.99	982.69

forecasting procedure in a negative manner which leads to less accurate prediction. Because of this, it is necessary to know how to deal with these noises, outliers, and null values in right way.

The collected data is containing a number of missing values as well as extreme values that vary over other data observations. This is an indication of measurement variability, errors in the experiment, or uniqueness. This problem is handled by giving the outliers null values to represent their position. The noise that is introduced into the dataset as a result of the missing data, has a detrimental impact on the competence of the model. The linear interpolation is used to fill in the blanks left by the missing data. It is a technique for generating new data points within a certain range of an existing discrete collection of data points that have already been determined. Other varieties of possible configurations of interpolation are linear, bi-linear, piece wise, polynomial, spline, cubic, and bi-cubic. Simply said, linear interpolation is the process of estimating a missing value by joining together dots in a straight line in ascending order in an efficient way (Huang, 2021). Due to its efficiency this study utilizes linear interpolation to replace missing values. The total dataset evaluated, includes 27,236 samples for every contaminant and meteorological parameter. Data pre-processing acts in accordance with the following steps:

2.2.1. Data imputation

The technique of substituting alternative values for missing data is known as data imputation (Zhang). When substituting a data point, the occurrences of missing values in the input parameters are thrown out. An impute function based on mean value technique is used to execute interpolation in order to make an approximation of the missing data in the case of the target object, which is the pollutant. Fig. 2 depicts the graphical representation of the heat map of the missing values. The x-axis of this figure is representing the air pollutants and considered meteorological parameters on the other side y -axis is representing the count of the missing values.

The measurement of the missing and non-missing values is given in Fig. 3. The blue part is representing the non-missing entries whereas the orange part says about the missing values.

The actual AQI value is represented graphically before applying the data normalization from the year 2015–2021 in Fig. 4:

2.2.2. Data aggregation

Data aggregation is the process of storing and representing data in a summary format (Clark and Avery, 1976). The data can come from more

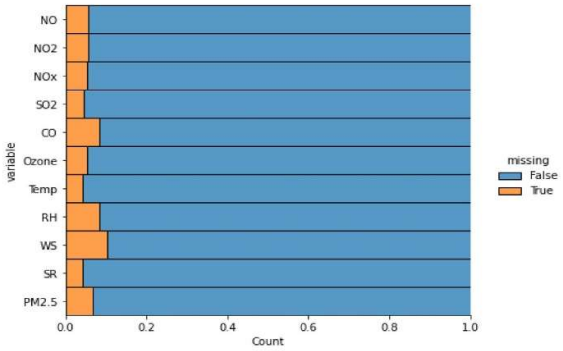


Fig. 3. Graphical representation of missing and non-missing values.

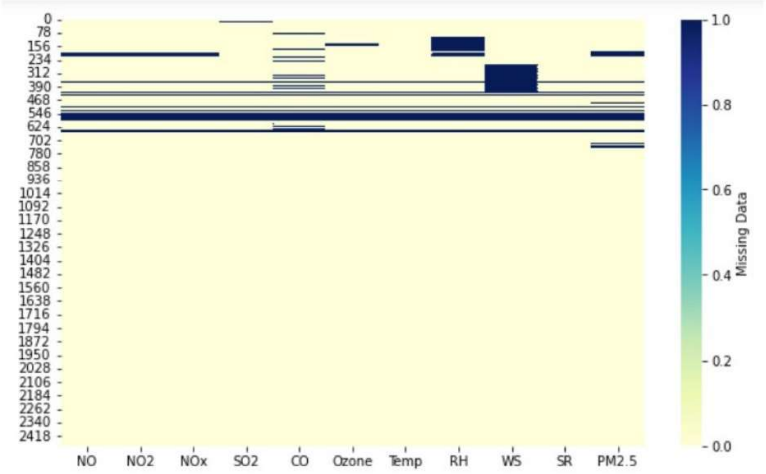


Fig. 2. Heat map of missing values.

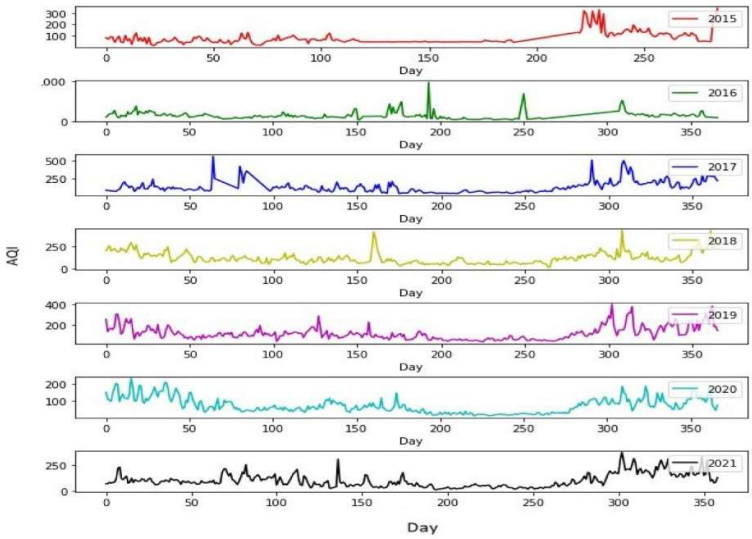


Fig. 4. AQI daily data for the years 2015–2021.

than one sources, and all of these sources can be put together in a data analysis description. This is a very important step because the accuracy of insights from data analysis depends a lot on the amount and quality of the data used. To get useful results, it is necessary to collect accurate, high-quality data in a large enough amount. Collecting data can help in everything-from making decisions about financing or business strategy, setting prices, running operations, and coming up with marketing plans etc.

2.2.3. Data normalization

Attributes consisting of several characteristics with different units, must be scaled to a specific range to ensure that each attribute receives the same amount of weightage. This process is known as data normalization (Turky et al., 2021). This prevents a potentially more significant trait from being overshadowed by a less significant one that has a wider range of values. In this case we re-scale all of the characteristics utilising the Z-score normalization or the mean - standard deviation scaling. The dataset has been normalized by the use of the mathematical formula, which can be written as:

$$X_{norm} = \frac{(X - X_{mean})}{X_{std}} \quad (1)$$

where, X_{norm} = value of attribute obtained after normalization.

X_{mean} = mean of the population and
X_{std} = standard deviation of the population.

AQI value is obtained after testing with normalized dataset which uses a minmaxscalar of range 0–1. The information shown in Table 4 makes it clear that the ranges of mean and standard deviation of our data are not the same. The gradient can fluctuate, and it may take a very long time to get either a local or a global minima since the ranges of possible values for the many attributes are not the same. Therefore, using Min-Max Normalization, the data are normalized on a scale in between 0 and 1 in order to tackle the issue of model learning. This ensures that the values of the various characteristics are not comparable to one another, which enables gradients to converge at a faster rate.

2.2.4. Feature selection

In the construction of a predictive model, one of the most important steps is feature selection (Agrawal and Sharma, 2022), which is the process of minimising the number of input variables or attributes by removing the irrelevant features. It is necessary to limit the number of parameters in order to enhance the performance of the model as well as to reduce the computational cost of modelling. An essential function that helps to reduce the number of input features in predictive modelling is "feature importance score" that provides information and insight of the working dataset. The term "feature importance" relates to the methods for scoring each input attribute for a certain model; the scores merely indicate the "importance" of each feature. A higher score indicates that the particular characteristic will have more impact on the model being used to forecast the target attribute. Here, the feature importance score is calculated based on permutation feature importance method. All features are arranged in ascending/descending order of its in permuted feature importance values.

Permutation feature importance is a technique for model inspection used for any of the fitted non linear estimator on a tabular form of data. The permutation feature importance refers to reduction of a model score for single feature values is shuffled randomly (Breiman, 2001). The process for calculation of permutation feature importance breaks the propinquity among the feature and the target in the tabular dataset. In this way it a technique to determine the dependency of the model on the feature. This technique can be used a number of times with various permutations of the feature of the dataset.

The algorithm to compute the permutation feature importance for each feature of a tabular dataset is given below as:

Algorithm 1
Algorithm to get Permutation Feature Importance of a feature

Require: Predictive Model: m, Dataset: D with J rows and P columns
1: Compute score 's' of the model 'm' on D (it is accuracy for a classifier and R^2 for a regressor)
2: **for** each feature p in D) **do**
3: **for** each repetition j in 1, ….,J **do**
4: Shuffle column 'p' randomly to generate a corrupted data $D_{j,p}$ for column 'p'
5: Compute score $s_{j,p}$ of 'm' on $D^{'}_{j,p}$.

(continued on next page)

Algorithm 1 (*continued*)

6: **end for**
7: Compute the feature importance i_p for feature f_p given as $i_p = s - \frac{1}{J}\sum_{j=1}^{J} s_{j,p}$ (2)
8: **end for**

The feature importance score of each attribute of developed dataset is depicted in Fig. 5. As $PM_{2.5}$ is one of the most prime ingredients in AQI calculation, that's why $PM_{2.5}$ must be included as a rich feature in the feature vector. So, in Fig. 5 the feature importance score of the remaining components excluding $PM_{2.5}$ is shown.

2.3. Machine learning and deep learning models

Various Machine Learning models, Deep Learning models and hybrid model are developed and obtained results from these developed models are compared mutually to get a statistical idea about the overall performance comparisons. The basic and brief idea about the used ML and DL models are given below:

2.3.1. Linear regression

In terms of data analytics (Ghani et al., 2019), Linear-Regression is the simplest and most often used ML approach. In linear regression, there is only one independent variable, and the relationship between the independent variable (x) and the dependent variable (y) is linear (Yu and Yao, 2017), (Pal and Bharati, 2019). The line can be modelled is mathematically expressed as follows (Bonaccorso, 2017):

Consider the dataset of real value input vector is given by X and $X = [\overrightarrow{x_1}, \overrightarrow{x_2}, \ldots\ldots, \overrightarrow{x_m}]$ and $\overrightarrow{x_i} \in \mathbb{R}^m$, where $\mathbb{R}$ is the set of real numbers.

Each of the input vector is mapped with a real valued output i.e. y_i. Let us consider, Y represents the set of real valued output mapped with every real valued input vector i.e. $Y = [y_1, y_2, \ldots\ldots, y_m]$, where $y_i \in \mathbb{R}$.

In linear regression it is tried to draw a line that best fits the points based on the given data.

$$y = a_0 + \sum_{i=1}^{m} a_i x_i \tag{3}$$

The best fit straight line is the red line shown in Fig. 6.

The objective of the linear regression method is to determine the optimal a_0 and a_i values. Where, a_0 is estimated intercept and a_i is the slope of the straight line.

2.3.2. K-Nearest Neighbor

A supervised machine learning technique called the K-Nearest Neighbor or KNN relies on the idea that how closely a point's value resembles the values of its neighbors. Therefore, it operates by selecting

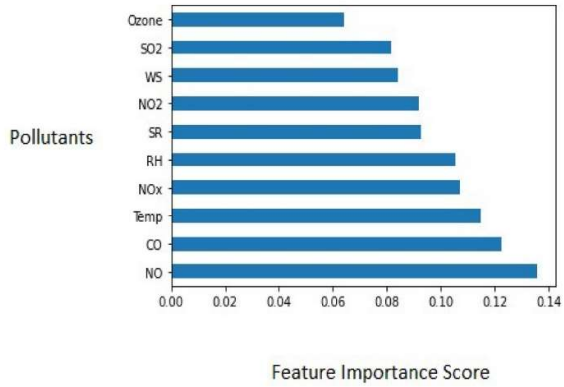


Fig. 5. Feature importance score.

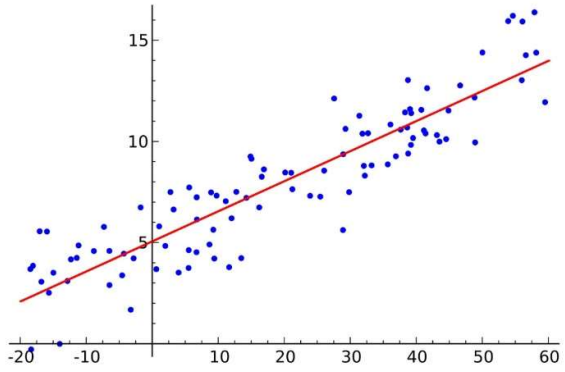


Fig. 6. Linear regression.

the K neighbor value that is nearest to the site of interest and selecting the class that appears the most frequently (Zhang et al., 2017a). The Euclidean distance (d) between two locations (a_1, a_2) and (b_1,b_2) may be calculated as follows (Zhang et al., 2017b):

$$d = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2} \tag{4}$$

For n-dimensional space, d can be obtained as:

$$d = \sum_{i=0}^{n} \sqrt{(a_i - b_i)^2} \tag{5}$$

The graphical representation of KNN is shown in Fig. 7.

2.3.3. Support vector machine

A nonlinear mapping φ_j (x) is used in SVM to translate data events x into a multidimensional feature space F, which then allows a linear regression model to be fitted. Again, a kernel function determines how input data is mapped further into new feature space. An interesting property of SVM is its approach to modelling error minimization, rather than focusing just on the apparent training errors (Fan et al., 2020).

In this case, the training dataset T is shown as:

$$T = (x_1, y_1), (x_2, y_2), \ldots\ldots, (x_n, y_n) \tag{6}$$

where $x \in X \subset \mathbb{R}^n$ are the inputs for training and $y \in Y \subset \mathbb{R}^n$ are the anticipated results of training.

A nonlinear function representing the SVM is given by equation (7):

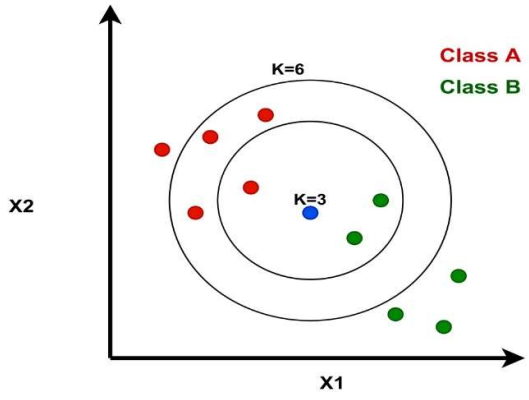


Fig. 7. K-nearest neighbor.

$$f(x) = w^T \varphi(x_i) + b \tag{7}$$

where, w denotes the vector of weights, bias is denoted by b, x is the input space, $\varphi(x_i)$ is the high-dimensional feature space. The aim is to develop a function f(x) that has the least possible deviation $\in$ from the goals y_i while fitting the training dataset T. Three training factors determine SVM's effectiveness and generalizability, these are:

- C (the regularization parameter)
- The kernel functions
- $\in$ (the insensitive zone)

Fig. 8 shows the graphical presentation of SVM.

2.3.4. Long Short-Term Memory

In the LSTM model, there are three different types of gates: input gates, forget gates, and output gates (Lei et al., 2019; Alvi et al., 2022). The output from the input modulator gate is fed into memory cell, which is responsible for the collection of any new information that is received from the outside world. The forget gate provides the following generation with instructions that which data are to be preserved and which data are to be thrown away. This is how it determines which delays are optimal for the data series that is provided. The output gate receives the outcomes of the computations in order to process them (Danihelka et al., 2016).

The structure of an LSTM cell is shown in Fig. 9:

A mathematical representation of the cells may be found as follows (Dey and Salem, 2017):

Input gate is represented as follows:

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \tag{8}$$

Forget gate is expressed by f_t:

$$f_t = \sigma_g\left(W_f x_t + U_f h_{t-1} + b_f\right) \tag{9}$$

Equation (10) depicts the mathematical expression for output gate. Output gate:

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \tag{10}$$

The Memory cell is represented as follow:

$$c_t = f_t \circ c_{(t-1)} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \tag{11}$$

Shadow state is given by h_t:

$$h_t = o_t \circ \sigma_h(c_t) \tag{12}$$

initial values $c_o = 0$ and $h_o = 0$ and the operator $\circ$ designates product in an element-wise manner.

The time stamp is represented by the subscript t.

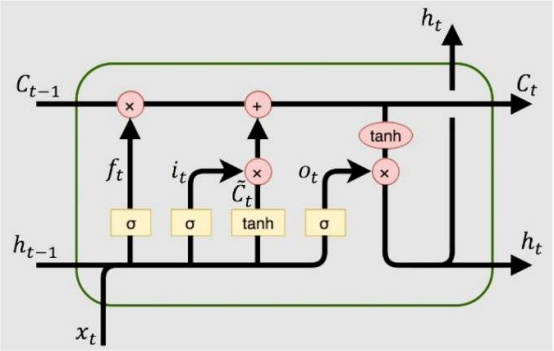


Fig. 9. Long short term memory cell.

$x_t \in R^d$: Vector to be used as input to LSTM unit.
$f_t \in R^h$: Activation vector used for Forget gate of LSTM unit.
$i_t \in R^h$: Activation vector used for Input gate of LSTM unit.
$o_t \in R^h$: Activation vector used for Output gate of LSTM unit.
$h_t \in R^h$: Output vector of LSTM unit.
$c_t \in R^h$: Vector to be used for cell state.
$W \in R^{(h_g d)}$, $U \in R^{(h_g h)}$, $b \in R^{(h)}$: Weight matrices and bias vector parameters that are trainable where superscripts $\in h$ and $\in d$ refer to the number of input features and number of hidden units respectively.
σ_g: Sigmoid function
σ_c: Hyperbolic tangent function
σ_h: Hyperbolic tangent function or identity function

A completely connected hidden layer is used to pass through the output from LSTM layers. The output layer produces the pollutant concentration at time (k+1), which is represented by $C^p_{(k+1)}$.

2.3.5. Gated Recurrent Unit

In addition to maintain the LSTM's properties, GRU significantly simplifies the structure of the algorithm. In the GRU model, there are numerous levels of interconnection (Tang et al., 2016) (Alvi et al., 2021). GRU's duplicate module structure is simpler than LSTM but both have recurrent neural networks with duplicate modules (Chen et al., 2019). Fig. 10 shows the GRU cell which is containing only two gates, the update gate (Z_t) and the reset gate (r_t).

A mathematical representation of the cell may be found as (Dey and Salem, 2017):

$$r_t = \sigma(W_r.[h_{t-1}, x_t]) \tag{13}$$

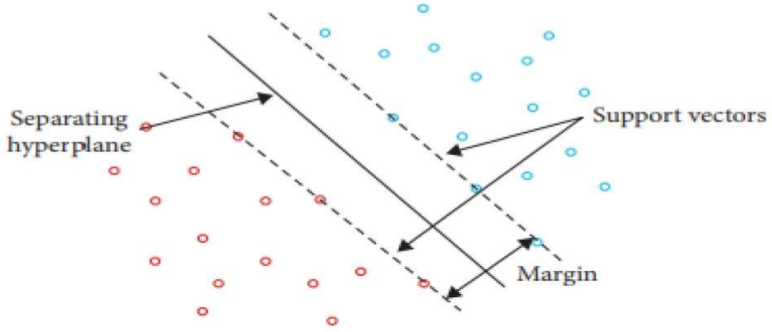


Fig. 8. Support vector machine.

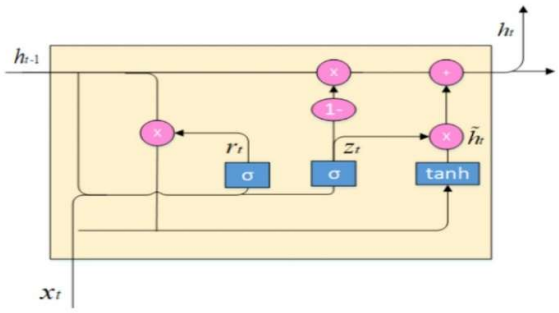


Fig. 10. Gated recurrent unit cell.

$$z_t = \sigma(W_z.[h_{t-1}, x_t]) \tag{14}$$

$$h_t = \tanh(W_h.[r_t * h_{t-1}, x_t]) \tag{15}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * h_t \tag{16}$$

$$y_t = \sigma(W_o.h_t) \tag{17}$$

Here, σ is the logarithmic sigmoid. The input value x and the preceding hidden state value h_t are both represented in these expressions. Learned weight matrices are represented by: W_r, W_z and W_h.

2.4. Performance evaluation metric

Performance evaluation metric is nothing but the benchmark for measuring the efficiency and efficacy of the ML and DL models. Model construction cannot be started until a comprehensive and in-depth evaluation has been completed. So, performance evaluation is one of the most important step for building a model and there are numerous metrics that can evaluate the performance of the ML and DL models. The performance evaluation metrics utilized in this work are enlisted below:

- **R-squared (R^2)**: R-squared (R^2) (Rights and Sterba, 2021) is a measure of the amount of variation in the result that can be accounted by the factors.
- **Mean Squared Error (MSE):** In mathematics, the Mean Squared Error (MSE) (Azeez and Adewoye) is the same thing as the root square of the mean squared error and the MSE is the lowest absolute disagreement between observed actual output values and predicted values by the model. The average squared difference between the actual output and the predicted output gives the MSE values which is represented by the following mathematical formula.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(P_{ACT(i)} - \mathrm{Q}_{\mathrm{PRED(i)}}\right)^2 \tag{18}$$

- **Root Mean Squared Error (RMSE):** Using the Root Mean Squared Error (RMSE) (Karunasingha, 2022), we may determine how accurate a model's prediction are in light of actual data. The RMSE is calculated by taking the square root of the MSE. The RMSE is the minimum absolute discrepancy between these two sets of values. Mathematically RMSE is represented as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(P_{ACT(i)} - \mathrm{Q}_{\mathrm{PRED(i)}}\right)^2} \tag{19}$$

- **Mean Absolute Error (MAE):** Mean Absolute Error (MAE) (Brassington, 2017) performs the same thing that RMSE does, which is to quantify the error in the model's prediction. MAE is computed by aggregating the absolute difference between the actual output and the predicted output of each observation over the entire dataset and then dividing the acquired sum by the total number of observations in the dataset. Equation (19) denotes the mathematical expression for MAE.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|P_{ACT(i)} - \mathrm{Q}_{\mathrm{PRED(i)}}\right| \tag{20}$$

Here, $P_{ACT(i)}$ is the actual output of the ith sample, $Q_{PRED(i)}$ is the predicted output of the ith sample, n is the total number of samples. The evaluation of MSE, RMSE and MAE is portrayed by the following example. Table 5 shows the values of actual and predicted output for three observations.

3. Implementation and result

The dataset is comprised of air quality data and meteorological data. Based on the empirical research the best results are obtained by partitioning the dataset into 70–80% for training and 20–30% for testing. That's why the dataset in this work is bisected as 80% for training and 20% for testing purposes. Different ML and DL models are trained with the training dataset, tested on testing dataset to evaluate the results for analysing the performance of the models.

Further subsections discuss the experimental setup, implementation and result analysis of the different ML and DL models such as Linear Regression model, K-Nearest Neighbor model, Support Vector Machine model, Long Short Term memory model, Gated Recurrent Unit model and finally the proposed hybrid LSTM-GRU model.

3.1. Experimental setup

The work presented in this article aims to develop a prediction model to predict the AQI value using ML and DL algorithms. The proposed approach is the amalgamation of the LSTM and GRU model. The simulation and modelling of this work were performed on a machine with an Intel Core i7-5500U CPU @ 2.40 GHz 2401 Mhz and 12 GB RAM running Microsoft Windows 10 Professional. All models for AQI forecasting are designed and developed using the Python programming environment created on the mentioned machine.

3.2. Implementation using linear regression model

The graph for AQI prediction with respect to time (day) for the actual data and the predicted data for linear regression model is shown in Fig. 11. The performance assessment of the Linear Regression model is presented using MAE, RMSE and R^2 methods. The obtained MAE, RMSE and R^2 values are 38.86, 57.09 and 0.42 respectively.

3.3. Implementation using K-Nearest Neighbor Model

The graph for AQI prediction with respect to time (day) for the actual data and the predicted data for K Nearest Neighbor Model with 10 neighbors is shown in Fig. 12. The performance assessment of the K-

Table 5
Actual vs. Predicted Output.

	Actual Output	Predicted Output
Observation 1	0.8	0.7
Observation 2	0.8	0.9
Observation 3	1.1	1.2

$$\mathrm{MSE} = \frac{1}{3}[(0.8-0.7)^2 + (0.8-0.9)^2 + (1.1-1.2)^2] = 0.01.$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{3}[(0.8-0.7)^2 + (0.8-0.9)^2 + (1.1-1.2)^2]} = 0.1.$$

$$\mathrm{MSE} = \frac{1}{3}|(0.8-0.7) + (0.8-0.9) + (1.1-1.2)| = 0.1.$$

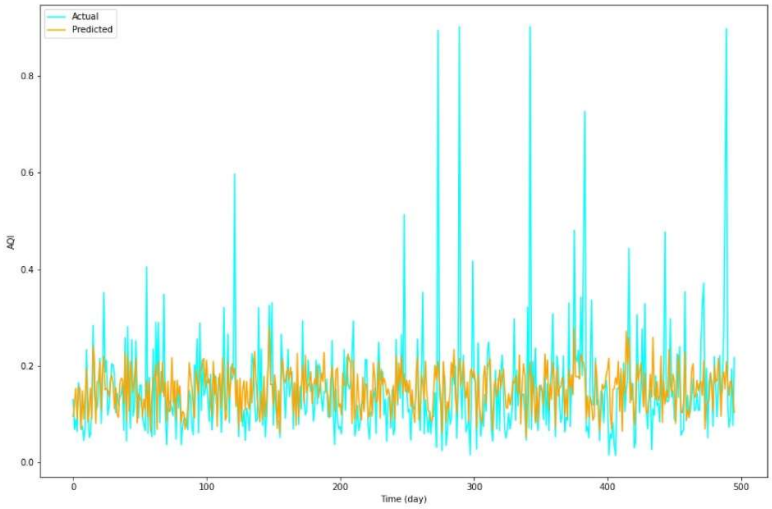


Fig. 11. Actual output Vs Predicted output for Linear Regression Model.

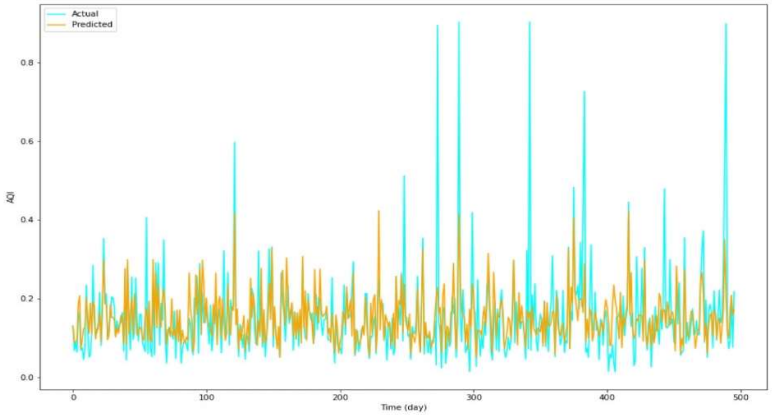


Fig. 12. Actual output Vs Predicted output for K-Nearest neighbor Model.

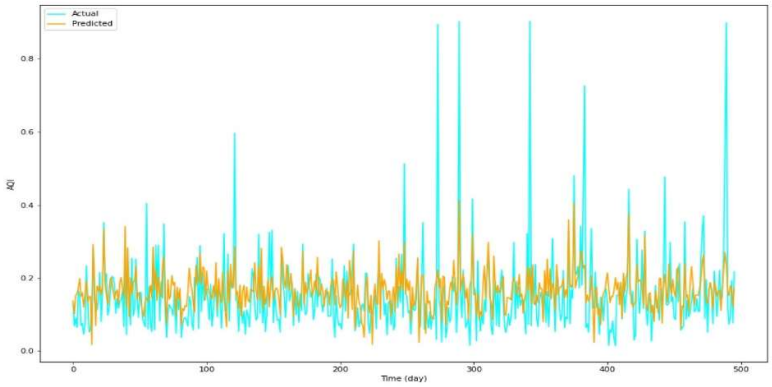


Fig. 13. Actual output Vs Predicted output for Support Vector Machine.

Nearest Neighbor model is presented using MAE, RMSE and R^2 methods. The obtained MAE, RMSE and R^2 values are 37.86, 57.65 and 0.47 respectively.

3.4. Implementation using support vector machine

A Support Vector Machine model has been constructed by utilising the Support Vector Regressor function and the Gaussian Radial Basis function (RBF) kernel parameter to the train dataset and generate predictions on the test dataset. The graph for AQI prediction with respect to time (day) for the actual data and the predicted data for SVM Model is shown in Fig. 13. The performance assessment of the SVM model is presented using MAE, RMSE and R^2 methods. The obtained MAE, RMSE and R^2 values are 49.09, 60.63 and 0.56 respectively.

3.5. Implementation using Long Short Term Memory

It is essential to scale the data before moving on to LSTM model for getting more accurate result. In order to maintain the same range of attribute variances throughout both the train dataset and the test dataset, Python's MinMaxScaler() method has been used. In the LSTM model 10 input feature vectors are fed into the input layer and one hidden layer are used with 50 neurons. The output is given by 1 target neuron using 'Relu' activation function. The number of batch size and epochs taken for the LSTM model are 16 and 10 respectively.

The graph for AQI prediction with respect to time (day) for the actual data and the predicted data for Long Short-Term Memory model on the testing samples is shown in Fig. 14. The performance assessment of LSTM model is presented using MAE, RMSE and R^2 methods. The obtained MAE, RMSE and R^2 values are 58.97, 61.37 and 0.78 respectively.

3.6. Implementation using Gated Recurrent Unit

The Gated Recurrent Unit (GRU) network is designed similar to LSTM for evaluation. It uses Min-Max function to scale data. At first the data are transmitted in a sequence. Then the dropout function has been applied to prevent data from being overfit and also a dense layer consisting of a single unit is utilized to produce a single variable output. The graph for the actual output and the predicted output for Gated Recurrent Unit model for AQI prediction with respect to time (day) on testing samples is shown in Fig. 15. The performance assessment of GRU model is presented using MAE, RMSE and R^2 methods. The obtained MAE, RMSE and R^2 values are 42.66, 49.45 and 0.65 respectively.

3.7. Implementation using LSTM-GRU model

In this work, a two-stage framework for foreseeing data on air quality is developed. When the process of input data preparation is started, first the method searches for any missing values and then uses the median function to replace them. Min-max scalar approaches are used for the datasets that have previously been normalized. Then, the GRU and LSTM deep learning models are integrated into a hybrid model.

To build up the LSTM-GRU integrated model, the data spanning from the seven years prior to the present is taken. In the proposed compound model, the LSTM layer is in charge of gathering the input samples and sending it on to the GRU layer, where a dropout function is used to prevent the data from being overfit. For the LSTM framework 10 input feature vectors are fed to the input nodes of the input layers which are connected with 100 neurons of the hidden layer using the 'tanh' activation function. A dropout with 0.2 is used to avoid the overfitting. Whereas, the GRU framework consists of a hidden layer with 50 neurons along with 'Relu' activation function and 0.2 dropout.

The graph for the actual output and the predicted output for LSTM-GRU model for AQI prediction with respect to time (day) on testing samples is shown in Fig. 16.

The performance assessment of LSTM-GRU model is presented using MAE, RMSE and R^2 methods. The obtained MAE, RMSE and R^2 values are 36.11, 52.21 and 0.84 respectively.

3.8. Flow diagram of the implemented integrated LSTM-GRU model

Fig. 17 depicts the work flow diagram of the proposed integrated LSTM-GRU Model.

It is depicted from this section that the proposed integrated LSTM-GRU model has the least MAE value and R^2 value while predicting the AQI based on the collected dataset from CPCB. The subsequent section discusses the comparative study on the basis of the performance metrics of the proposed integrated LSTM-GRU model with the other existing ML and DL models.

4. Comparative analysis of performance of different models

This section discusses the comparison of evaluation metrics of proposed integrated LSTM-GRU model with other ML and DL models. Evaluation is a crucial step for any model implementation and that also allows to identify the optimal model among different models based on the performance result. In this study the evaluation analysis or the comparative analysis of the performance of different models are conducted utilising three metrics. In order to conclude the calculations, the actual value was compared to the anticipated outcomes. In the context of

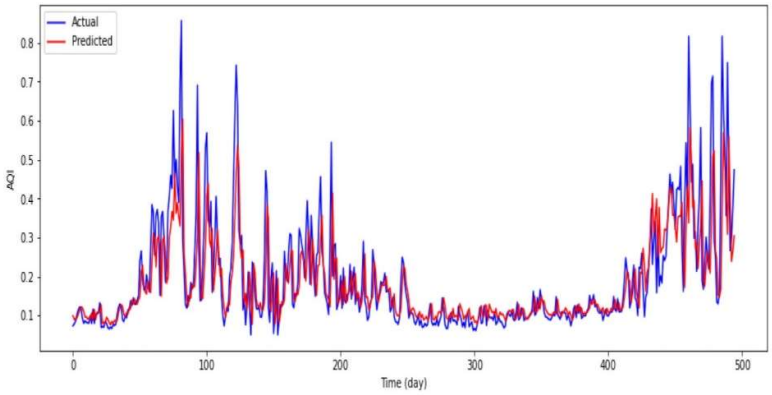


Fig. 14. Actual output Vs Predicted output for Long Short Term Memory.

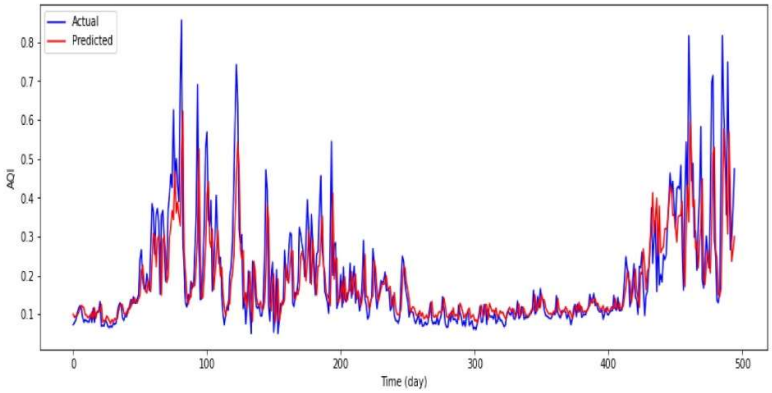


Fig. 15. Actual output Vs Predicted output for Gated Recurrent Unit.

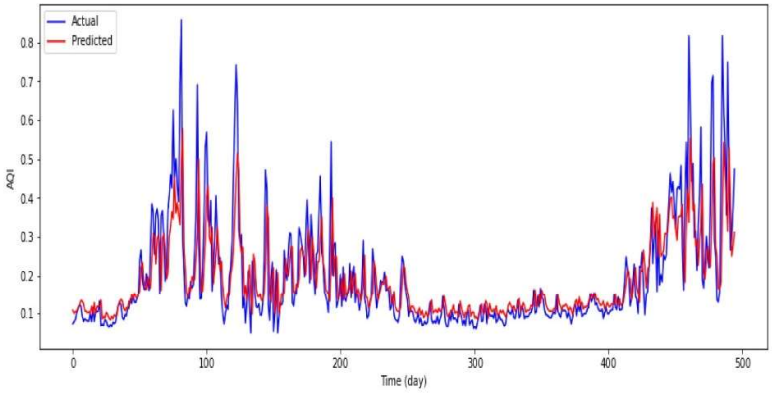


Fig. 16. Actual output Vs Predicted output for LSTM-GRU Model.

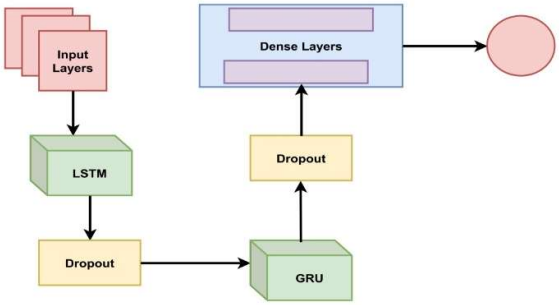


Fig. 17. LSTM-GRU model.

this investigation, RMSE, R^2, and MAE are applied as metrics to assess how well the models are performing.

The optimal result that are obtained after carefully evaluating with various ratios of training-testing data and comparison of the scores of the models as shown in Table 6. The dataset is divided into 80 percent of training data and 20 percent testing data for the purposes of evaluation. This decision is made due to the fact that the optimal result has been obtained after carefully evaluating with different ratios of training-testing data.

Table 7 is representing the metrics wise performance comparison of the different models.

Using the accuracy metric as a point of comparison in Table 7, it has been shown that the MAE 36.11 of the suggested model is lower than that of other models. Hence, it is reasonable to assume that the proposed model is appropriate for projecting data related to air quality. The suggested model, on the other hand, has an R^2 of 0.84, while LSTM has an R^2 of 0.78. This indicates that LSTM-GRU performs better than the LSTM model in this particular scenario. When compared to other models, some measurements provide less favourable findings.

The result of the proposed model is compared with the other existing approaches and the corresponding comparative study is reported in the following table.

5. Conclusion and future scope

A precise and steady AQI forecasting is not only essential for promoting urban public health, but it is also important for maintainable development of environment under the adverse effects of air pollution. This research proposes a hybrid LSTM-GRU model for predicting the AQI in a very polluted city. The performance of the LSTM-GRU model is compared with the other standalone ML and DL models' performance in terms of R^2, RMSE and MAE parameters. The results manifest that the developed hybrid model produces less mistakes than stand-alone models, indicating its superiority. The proposed method can also be extended to predict AQI of other cities jointly for which we must require

Table 6
Performance Comparison based on training validation test split method.

Model	Metrics	60:40	65:35	70:30	75:25	80:20	85:15	90:10	95:05
LR	R^2	0.37	0.38	0.37	0.42	0.40	0.38	0.40	0.42
	MAE	38.87	38.87	38.87	38.86	38.86	38.86	38.86	38.86
	RMSE	61.01	61.20	60.87	57.09	59.22	61.38	56.26	55.01
KNN	R^2	0.42	0.43	0.41	0.47	0.48	0.49	0.52	0.52
	MAE	37.86	37.86	37.86	37.86	37.86	37.86	37.86	37.86
	RMSE	54.90	54.97	55.28	50.65	51.63	52.13	46.66	46.50
SVM	R^2	0.57	0.54	0.54	0.56	0.58	0.58	0.51	0.51
	MAE	49.09	49.10	49.09	49.09	49.09	49.09	49.09	49.10
	RMSE	62.19	64.02	63.64	60.63	61.39	62.59	61.01	60.86
LSTM	R^2	0.73	0.71	0.73	0.78	0.78	0.81	0.79	0.81
	MAE	58.97	58.98	58.98	58.97	58.97	58.97	58.97	58.97
	RMSE	61.37	61.37	61.37	61.37	61.37	61.37	61.37	61.37
GRU	R^2	0.61	0.61	0.63	0.64	0.66	0.64	0.63	0.66
	MAE	42.67	42.67	42.67	42.66	42.66	42.66	42.66	42.66
	RMSE	54.21	54.87	53.52	49.45	50.73	51.25	47.95	46.85
LSTM-GRU	R^2	0.69	0.70	0.73	0.75	0.84	0.72	0.73	0.75
	MAE	36.12	36.11	36.12	36.11	36.11	36.11	36.11	36.11
	RMSE	57.77	57.91	54.33	52.21	52.03	54.43	52.21	51.36

Table 7
Performance comparison.

Model	R^2	MAE	RMSE
Proposed Model	0.84	36.11	52.03
LR	0.40	38.86	59.22
KNN	0.48	37.86	51.63
SVM	0.58	49.09	61.39
LSTM	0.78	58.97	61.37
GRU	0.66	42.66	50.73

Table 8
Comparison of the proposed LSTM-GRU model with other existing approaches.

Sl. No.	Ref.	Classifier	R^2-value
1	Srivastava et al. (2018)	LR, SDG, RF, DT, SVM, CNN, GB,	R^2 = 0.65646, 0.65922, 0.67, 0.62, 0.69275,0.68478, 0.69647, 0.69275
2	Alireza et al. (2021)	HSD, HTPD, CEEMDAN-ELM, CEEMDAN-GRNN	R^2 = 0.74
3	Proposed Approach	LSTM-GRU	R^2 = 0.84

high performance machine along with GPU. In future a large dataset with many features may be created by taking data from different areas for enhancing the experiments with different feature creation and feature selection techniques and classification techniques such as ensemble technique, fuzzy logic techniques etc. with different evaluation metrics such as accuracy, precision, recall etc. where model hyperparameter optimization can be explored and exploited.

Author statement

Nairita Sarkar: Writing – original draft preparation Conceptualization. Rajan Gupta: Methodology, Software and Data curation. Dr.Pankaj Kumar Keserwani: Supervision. Prof. Mahesh Chandra Govil: Visualization Writing- Reviewing and Editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data source is cited.

References

Agrawal, S., Sharma, D.K., 2022. Feature extraction and selection techniques for time series data classification: a comparative analysis. In: 2022 9th International Conference on Computing for Sustainable Global Development (INDIACom). IEEE, pp. 860–865.

Alireza, R., Jamil, A., Tzanis, C.G., 2021. Air quality data series estimation based on machine learning approaches for urban environments. Air Q. Atmos. Health 14 (2), 191–201.

Alvi, A.M., Siuly, S., Wang, H., 2021. Developing a deep learning based approach for anomalies detection from eeg data. In: International Conference on Web Information Systems Engineering. Springer, pp. 591–602.

O. I. Azeez, K. B. Adewoye, Mean Square Error in Ml Estimation of Two-Level Time Series Models.

Alvi, A.M., Siuly, S., Wang, H., 2022. A long short-term memory based framework for early detection of mild cognitive impairment from eeg signals', in. IEEE Trans. Emerg. Topics Comput. Intell. https://doi.org/10.1109/TETCI.2022.3186180.

Bhalgat, P., Bhoite, S., Pitare, S., 2019. Air quality prediction using machine learning algorithms. Int. J. Comput. Appl. Technol. Res. 8 (9), 367–390.

Bonaccorso, G., 2017. Machine Learning Algorithms. Packt Publishing Ltd.

Brassington, G., 2017. Mean absolute error and root mean square error: which is the better metric for assessing model performance?. In: EGU General Assembly Conference Abstracts, p. 3574.

Breiman, L., 2001. Random forests. Mach. Learn. 45 (1), 5–32.

Campbell-Lendrum, D., Prüss-Ustün, A., 2019. Climate change, air pollution and noncommunicable diseases. Bull. World Health Organ. 97 (2), 160.

Castelli, M., Clemente, F.M., Popovič, A., Silva, S., Vanneschi, L., 2020. A Machine Learning Approach to Predict Air Quality in california. Complexity.

Chen, J., Jing, H., Chang, Y., Liu, Q., 2019. Gated recurrent unit based recurrent neural network for remaining useful life prediction of nonlinear deterioration process. Reliab. Eng. Syst. Saf. 185, 372–382.

S. Chen, G. Kan, J. Li, K. Liang, Y. Hong, Investigating China's urban air quality using big data, information theory, and machine learning., Pol. J. Environ. Stud. 27 (2).

Clark, W.A., Avery, K.L., 1976. The effects of data aggregation in statistical analysis. Geogr. Anal. 8 (4), 428–438.

Danihelka, I., Wayne, G., Uria, B., Kalchbrenner, N., Graves, A., 2016. Associative long short-term memory. In: International Conference on Machine Learning. PMLR, pp. 1986–1994.

Dey, R., Salem, F.M., 2017. Gate-variants of gated recurrent unit (gru) neural networks. In: 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS). IEEE, pp. 1597–1600.

Fan, J., Wu, L., Ma, X., Zhou, H., Zhang, F., 2020. Hybrid support vector machines with heuristic algorithms for prediction of daily diffuse solar radiation in air-polluted regions. Renew. Energy 145, 2034–2045.

Ghani, N.A., Hamid, S., Hashem, I.A.T., Ahmed, E., 2019. Social media big data analytics: a survey. Comput. Hum. Behav. 101, 417–428.

Glencross, D.A., Ho, T.-R., Camina, N., Hawrylowicz, C.M., Pfeffer, P.E., 2020. Air pollution and its effects on the immune system. Free Radic. Biol. Med. 151, 56–68.

Gocheva-Ilieva, S.G., Ivanov, A.V., Voynikova, D.S., Boyadzhiev, D.T., 2014. Time series analysis and forecasting for air pollution in small urban area: an sarima and factor analysis approach. Stoch. Environ. Res. Risk Assess. 28 (4), 1045–1060.

Huang, G., 2021. Missing data filling method based on linear interpolation and lightgbm. In: Journal of Physics: Conference Series, vol. 1754. IOP Publishing, 012187.

Iskandaryan, D., Ramos, F., Trilles, S., 2020. Air quality prediction in smart cities using machine learning technologies based on sensor data: a review. Appl. Sci. 10 (7), 2401.
Janarthanan, R., Partheeban, P., Somasundaram, K., Elamparithi, P.N., 2021. A deep learning approach for prediction of air quality index in a metropolitan city. Sustain. Cities Soc. 67, 102720.
Karunasingha, D.S.K., 2022. Root mean square error or mean absolute error? use their ratio as well. Inf. Sci. 585, 609–629.
Kleine Deters, J., Zalakeviciute, R., Gonzalez, M., Rybarczyk, Y., 2017. Modeling pm2. 5 urban pollution using machine learning and selected meteorological parameters. J. Electr. Comput. Eng.
Lei, J., Liu, C., Jiang, D., 2019. Fault diagnosis of wind turbine based on long short-term memory networks. Renew. Energy 133, 422–432.
Li, J., Li, X., Wang, K., 2019. Atmospheric pm2. 5 concentration prediction based on time series and interactive multiple model approach. Adv. Meteorol.
Li, G., Tang, Y., Yang, H., 2022. A new hybrid prediction model of air quality index based on secondary decomposition and improved kernel extreme learning machine. Chemosphere 305, 135348.
Liu, T., Lau, A.K., Sandbrink, K., Fung, J.C., 2018. Time series forecasting of air quality based on regional numerical modeling in Hong Kong. J. Geophys. Res. Atmos. 123 (8), 4175–4196.
Liu, H., Li, Q., Yu, D., Gu, Y., 2019. Air quality index and air pollutant concentration prediction based on machine learning algorithms. Appl. Sci. 9 (19), 4069.
Londhe, M., 2021. Data mining and machine learning approach for air quality index prediction. Int. J. Eng. Appl. Phys. 1 (2), 136–153.
Mahalingam, U., Elangovan, K., Dobhal, H., Valliappa, C., Shrestha, S., Kedam, G., 2019. A machine learning model for air quality prediction for smart cities. In: 2019 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET). IEEE, pp. 452–457.
Nigam, S., Rao, B., Kumar, N., Mhaisalkar, V., 2015. Air quality index-a comparative study for assessing the status of air quality. Res. J. Eng. Technol. 6 (2), 267–274.
Pal, M., Bharati, P., 2019. Introduction to correlation and linear regression analysis. In: Applications of Regression Techniques. Springer, pp. 1–18.
Pant, A., Sharma, S., Bansal, M., Narang, M., 2022. Comparative analysis of supervised machine learning techniques for aqi prediction. In: 2022 International Conference on Advanced Computing Technologies and Applications (ICACTA). IEEE, pp. 1–4.
P. Partheeban, Application of lstm models in predicting particulate matter (pm2. 5) levels for urban area, J. Eng. Res..
Pozzer, A., Dominici, F., Haines, A., Witt, C., Münzel, T., Lelieveld, J., 2020. Regional and global contributions of air pollution to risk of death from covid-19. Cardiovasc. Res. 116 (14), 2247–2253.
Rights, J.D., Sterba, S.K., 2021. R-Squared Measures for Multilevel Models with Three or More Levels. Multivariate Behavioral Research, pp. 1–28.
Sigamani, S., Venkatesan, R., 2022. Air quality index prediction with influence of meteorological parameters using machine learning model for iot application. Arabian J. Geosci. 15 (4), 1–12.
Singh, R.P., Chauhan, A., 2020. Impact of lockdown on air quality in India during covid-19 pandemic. Air Qual. Atmos. Health 13 (8), 921–928.
Srivastava, C., Singh, S., Singh, A.P., 2018. Estimation of air pollution in Delhi using machine learning techniques. In: 2018 International Conference on Computing, Power and Communication Technologies (GUCON). IEEE, pp. 304–309.
Tang, Y., Huang, Y., Wu, Z., Meng, H., Xu, M., Cai, L., 2016. Question detection from acoustic features using recurrent neural network with gated recurrent unit. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, pp. 6125–6129.
Turky, S.N., Al-Jumaili, A.S.A., Hasoun, R.K., 2021. Deep learning based on different methods for text summary: a survey. J. Al-Qadisiyah Comput. Sci. Math. 13 (1). Page–26.
Van, N., Van Thanh, P., Tran, D., Tran, D.-T., 2022. A new model of air quality prediction using lightweight machine learning. Int. J. Environ. Sci. Technol. 1–12.
Wu, Q., Lin, H., 2019. A novel optimal-hybrid model for daily air quality index prediction considering air pollutant factors. Sci. Total Environ. 683, 808–821.
Yu, C., Yao, W., 2017. Robust linear regression: a review and comparison. Commun. Stat. Simulat. Comput. 46 (8), 6261–6282.
Yuan, Y., Liu, M., 2014. Discussion on the difference between air quality index (aqi) and air pollution index (api) j. Guangzhou Chem. Ind. 42, 164–166.
Z. Zhang, Missing data imputation: focusing on single imputation, Ann. Transl. Med. 4 (1).
Zhang, S., Li, X., Zong, M., Zhu, X., Wang, R., 2017a. Efficient knn classification with different numbers of nearest neighbors. IEEE Transact. Neural Networks Learn. Syst. 29 (5), 1774–1785.
Zhang, S., Li, X., Zong, M., Zhu, X., Cheng, D., 2017b. Learning k for knn classification. ACM Trans. Intell. Syst. Technol. (TIST) 8 (3), 1–19.
Zhou, X., Xu, J., Zeng, P., Meng, X., 2019. Air pollutant concentration prediction based on gru method. In: Journal of Physics: Conference Series, vol. 1168. IOP Publishing, 032058.
Zhu, S., Lian, X., Liu, H., Hu, J., Wang, Y., Che, J., 2017. Daily air quality index forecasting with hybrid models: a case in China. Environ. Pollut. 231, 1232–1244.
Zhu, J., Wu, P., Chen, H., Zhou, L., Tao, Z., 2018. A hybrid forecasting approach to air quality time series based on endpoint condition and combined forecasting model. Int. J. Environ. Res. Publ. Health 15 (9), 1941.
Zou, B., You, J., Lin, Y., Duan, X., Zhao, X., Fang, X., Campen, M.J., Li, S., 2019. Air pollution intervention and life-saving effect in China. Environ. Int. 125, 529–541.