Air Pollution Monitoring System using Stacked Attentional Vectormap Convolutional Bidirectional Network with Bobcat Optimization and IoT-Cloud

Prajul Mohandas [1], Subramanian .P[2, *], Surendran Rajendran[3]

[1,2,3] Department of Computer Science and Engineering, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, 602105, India.

[1] prajulmohandas@gmail.com

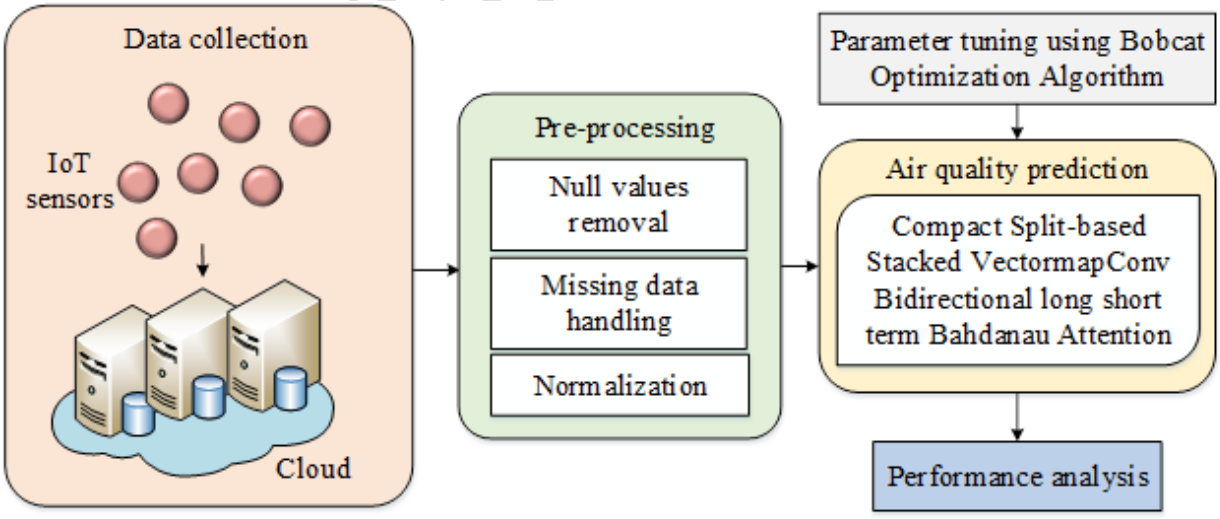
[2] subramanianp.sse@saveetha.com

[3] surendranr.sse@saveetha.com

[*]Corresponding author:

E-mail: subramanianp.sse@saveetha.com

GRAPHICAL ABSTRACT



ABSTRACT

Air pollution is one of the serious environmental problem that has an impact on ecosystems and human health worldwide. The prediction of air pollution can provide significant information that will permit all parties to take right initiatives. Predicting air quality is considered as a popular research area. A potential solution to air pollution has been suggested through the application of numerous time-series and artificial intelligence (AI) methods. These models are used with Internet of Things (IoT) devices in a cloud environment to forecast the air quality parameters. However, there exist several challenges such as overfitting issues, inaccurate real-time updates, and low precision. In this paper, an IoT cloud-based Air Pollution Monitoring System using Deep Learning (IoT-CAPM-DL) model under various meteorological situations is proposed to address the existing problems. Data collected from sensors are preprocessed to enhance its quality by eliminating null values, handling missing data, and normalization. Then, a robust Compact Split-based Stacked VectormapConv Bidirectional long short term Bahdanau Attention (CSplitStack-VBA) network is used to predict air quality parameters. A Bobcat Optimization Algorithm (BcOA) is used to tune the hyper-parameters of prediction model. The entire implementation is carried out using the Python platform and different kind of performance measures are calculated. The finding shows that the IoT-CAPM-DL model attains better MAE and RMSE value of 0.0076 and 0.0051. Thus, the experimental outcomes prove that the IoT-CAPM-DL model performed better in the prescribed dataset and produced significant results than existing approaches.

Keywords: Air pollution, Internet of Things, Vector map convolution, Bahdanau attention, Bobcat optimization algorithm, Cloud environment

1. Introduction

Over the past years, the world has become more intelligent and increasingly connected with the expansion of Internet of Things (IoT). Idrees et al. (2020), Liao et al. (2020), and Ullo et al. (2020)

published work related to the IoT sensor based air pollution predictor. IoT is deliberated as a wireless network of intelligent sensors that has the capability to collect and transmit data through embedded devices. The five main parts of an IoT devices are normally a processor, sensors, memory module, communications module, and power supply. Singh et al. (2021) discussed a gateway connects the sensors; it enables communication between the sensors and offers processing and storage abilities. The gateway can be hosted on edge or in the cloud. Time-series data generation from IoT devices, comprising robotics, sensors, and machines, is gaining popularity. Rapid data generation is a result of practical applications like air pollution monitoring.

For further analysis, the data are sent to a cloud or edge processing center. Owing to the detrimental effects on human health, air pollution has gained greater attention globally. As a result, it has become increasingly significant to monitor and forecast the air quality around people in the past few years. IoT is broadly applied in various fields to improve human health by connecting various sensors in diverse locations. Feng et al. (2024) explained air pollution is a major global concern and has numerous detrimental health effects. The World Health Organization (WHO) has estimated that ambient air pollution approximately caused over 7 million deaths worldwide in 2019, which is greater than 15% of all deaths expressed by Maio, S et al. (2023).

Wu et al. (2023) analyzed several areas have set up networks for monitoring air quality in real-time and combined a significant amount of monitoring data, which offers the fundamental data desirable for precise air pollutant prediction. Shin et al. (2023) derived the temporal dependencies still make it challenging to forecast the concentration of pollutants in a city. At any time interval, both the nearby and far-off historical time intervals and external causes have an influence on the pollution levels. Consequently, in order to offer an accurate prediction of air quality, an efficient model that fully extract and learn space and time dependencies as well as external inputs is

essential. Knowledge-driven and data-driven methods are the two general categories used to predict air quality.

Knowledge-driven models are capable to accurately assess the concentration of air pollutants since they combine both chemical and physical mechanisms to stimulate the processes of pollutant emission, diffusion, transformation, and transportation. Yet, these models are not supportive for the inspection of air quality issues at microscales, like in metropolitan regions. Because, they strongly depend on setting parameters that are hard to achieve and necessitate prior knowledge. Accordingly, data-driven techniques deliver an alternate method for accomplishing the predictive examination of the pollutant concentration in future by Liu et al. (2022). In contrast to knowledge-driven models, these models do not necessitate prior knowledge and are merely based on correlations between dependent variables and pollutant concentration data.

Shallow machine learning (ML), statistical, and deep learning (DL) techniques are the three subcategories of data-driven models by Maltare et al. (2023), and Ahmed et al. (2024). Time-series analysis is utilized by statistical models to predict future values based on the historical observed data. These models are efficient in dealing linear features however they are incompetent in capturing complex non-linear features. In contrast to shallow ML models, DL models have the capability to automatically identify important features and take unprocessed data as input for end-to-end prediction by Zhang et al. (2024), Prado-Rujas et al. (2024). DL architectures, a type of ML, have proven state-of-the-art outcomes in broad environmental prediction problems because of their strong generalization, potential non-linear mapping capabilities, and flexible model structure. Recurrent neural networks (RNNs) provides great benefit in handling with sequence learning challenges by Saravanan, D et al. (2023).

Thereby, Liu et al. (2023) exposed to acquire the temporal dynamics in pollutant sequence, RNNs and their variants, including bidirectional LSTM (BiLSTM) and long short-term memory networks (LSTM) networks are presented. But, the limited utilization of the spatial relationships within the monitoring network by RNN-based models has potentially obstructed their capability to process spatiotemporal data. Encouraged by convolutional neural networks (CNNs) potential for extracting spatial features, it has turn out to be dominant to utilize CNNs and RNNs to predict air quality. The most recent successor of CNN by Wang et al. (2024) is residual neural network (ResNet), which permits longer structures for learning deep abstract spatial relationships. Nevertheless, this model developed by Shaban et al. (2024) may need a considerable amount of inference time in order to handle new data. With the intension of better optimizing urban atmospheric forecasting, this research Motivates to develop a hybrid DL model based on the IoT Cloud in air pollutant monitoring network. The proposed model offers decision makers with correct and timely information on air quality trends by using cloud computing and DL. The main aim of the proposed work is

- To develop an IoT cloud-based Air Pollution Monitoring System using Deep Learning (IoT-CAPM-DL) under various meteorological situations for predicting the air pollution.

The Objectives of the proposed work is provided below as follows:

- To employ a Compact Split-based Stacked VectormapConv Bidirectional long short term Bahdanau Attention (CSplitStack-VBA) network, which combines compact split-attention (CSplitA), vectormap CNN, and stacked BiLSTM, and Bahdanau attention for extracting features and forecasting the air quality parameters.
- To tune the hyper-parameters of prediction model using a bio-inspired metaheuristic Bobcat Optimization Algorithm (BcOA) for minimizing the error rate.

- To validate the working of IoT-CAPM-DL model by comparing with state-of-the-art methods.

The scope of the proposed work is allows to make well-informed decisions that lowers the levels of air pollution and meets the air quality standards. The rest of the paper is aligned as follows. Section 2 deliberates the works related to the research proposed. Section 3 introduces the proposed IoT-CAPM-DL model. The performance of the IoT-CAPM-DL model for air quality prediction is assessed with simulated results in Section 4, and finally, the paper concludes and establish future directions in Section 5.

2. Related works

Bhushankumar Nemade and Deven Shah et al. (2022) suggested an IoT-based air pollution prediction system based on deep learning modified neural network (DLMNN) classifier. First, the H-ANFIS algorithm was utilized in the sensor nodes to identify the problematic node. MPCA algorithm was employed to extract features from the sensed data and remove unnecessary features. Next, the data was balanced using Entropy-HOA, and pre-processed using HDFS and replacement of missing attribute. After that, the pre-processed data was provided to DLMNN classifier, which could optimize the weight using pity beetle algorithm (PBA) for prediction. However, the complexity of this model was higher.

Shilpa Sonawani and Kailas Patil et al. (2024) recommend an IoT-based air quality monitoring, warning, and prediction system that could perceive indoor air quality parameters such as CO, NO2, PM2.5, NH3, O3, pressure, temperature, etc. Here, the multiheaded CNN-gated recurrent unit could identify the pollution concentration for upcoming hour. Moreover, the model employed a transfer learning (TL) approach if there was a limited availability of data for prediction. The findings showed that the performance enhancement of 55.42% had attained for prediction with

insufficient data. However, the overfitting problem could minimize the generalization ability of this model.

A combined air-quality prediction model based on the ARIMA-CNN-LSTM with dung beetle optimizer (DBO) was introduced by Jiahui Duan et al. (2023). To evade the blinding issue in hyperparameter setting of CNN-LSTM, ARIMA model was initially used to fit the linear portion of the data. Next, DBO was used to find the CNN-LSTM model's hyperparameters. As a result, the four cities had corresponding root mean square value (RMSE) values of 7.594, 14.94, 7.841 and 5.496; and R2 values of 0.989, 0.962, 0.953 and 0.953. However, this model needs to consider different influencing factors to enhance the model performance.

Subramanian Deepan & Murugan Saravanan et al. (2024) suggested using seasonal autoregressive integrated moving average (SARIMA) transductive LSTM (TLSTM) for air quality index prediction. In order to predict values that characterize historical trends regarded as seasonal patterns, a SARIMA model was employed. Furthermore, a TLSTM model acquired long-term dependencies for predicting air quality index by learning dependencies through recurrent memory blocks. Further, the TLSTM had maximized the accuracy close to test sites. The experimental results showed that the SARIMA-TLSTM model had accomplished a greater accuracy of 93%. However, the time complexity of this model was higher.

Shelly Sachdeva et al. (2024) suggested an integrated approach for predicting the air quality index using meteorological data and pollutant concentration. There were four modules in the framework for predicting air pollution. The first module had forecasted the concentrations of hazardous gases and particulate matter pollutants. Based on the historical air quality index data and pollutant data, the second module had predicted the air quality. By meteorological data and other data, the third module estimated air quality index. The output of second and third module were combined in the

fourth module to compute air quality index. The findings showed that the mean absolute error (MAE) for prediction was only 7.09. However, the model would need to reduce the computation complexity and improve the performance in terms of different performance indicators.

Periasamy, S., et al. (2024) developed a system based on transfer learning and quality indicators in recurrent network that is lightweight and has a skip connection to find the quality of air. By including skip contacts between the decoder and the linear forecasting layer, the suggested model lessens the decoding load.This study need to improve with larger datasets and more parameters will be added. Sundarapandi, A.M.S. et al. (2024) this study introduces a new prediction technique that combines a tree structural simple recurrent unit (LDTSRU) with a light weighted dense network. The input meteorological variables are first converted into grayscale images using a lightweighted dense network, which then looks for any noteworthy patterns within the variables.

Problem statement: Forecasting the concentrations of air pollutants at a specific location and time is a primary challenge. Several egression or classification has been used to predict concentration or categorize air quality. One significant problem is the inaccuracy in prediction since the conventional techniques struggle to capture nonlinear, complex relationships in pollution data, thereby tending to discrepancies between actual and predicted air quality levels. Several traditional methods rely on static model that cannot adapt to varying environmental conditions, or new data, which minimize their responsiveness to real-time variation in air quality. Furthermore, advanced machine learning and artificial intelligence methods are not often used to their full potential to improve forecast accuracy. Also, the high computational burden particularly of advanced approaches also limits their use especially in low-resource areas. To fully exploit the immense potential of air quality forecast in smart cities, a number of challenges, including model

interpretability, data accuracy, and the requirement for continuous model improvement should be addressed.

3. Proposed methodology

The potent tool that can assist businesses, governments, and individuals in managing and monitoring the quality of the air in their daily lives is an IOT- cloud based air pollution monitoring system. This system measures air pollutants using sensors and sends real-time data to a cloud server. Then, the data are analysis and visualization in accordance with environmental standards, and the application is allowed to remotely monitor the quality of air. The IoT-CAPM-DL model has considered various meteorological situations and it is modelled to measure several air quality parameters and pollutants in real time. Sensor nodes (SNs), WiFi module, gateway and cloud are used in the proposed system. The sensor nodes are positioned at various locations to collect information on the quality of the air, and wireless communication is utilized to transfer the data to the gateway. The SNs' data is aggregated via the gateway and sent to the cloud server for further processing and analysis. After storing the collected data in cloud, the raw data undergoes pre-processing to enhance its quality by eliminating null values, handling missing data, and normalizing data using L2-standardization. Then, a robust Compact Split-based Stacked VectormapConv Bidirectional long short term Bahdanau Attention (CSplitStack-BA) network is developed for extracting features and forecasting the air quality parameters. In addition, the hyper-parameters of CSplitStack-VBA network can be tuned using Bobcat Optimization Algorithm (BcOA) to enhance the forecasting performance. The block diagram of IoT-CAPM-DL model is given in Figure 1.

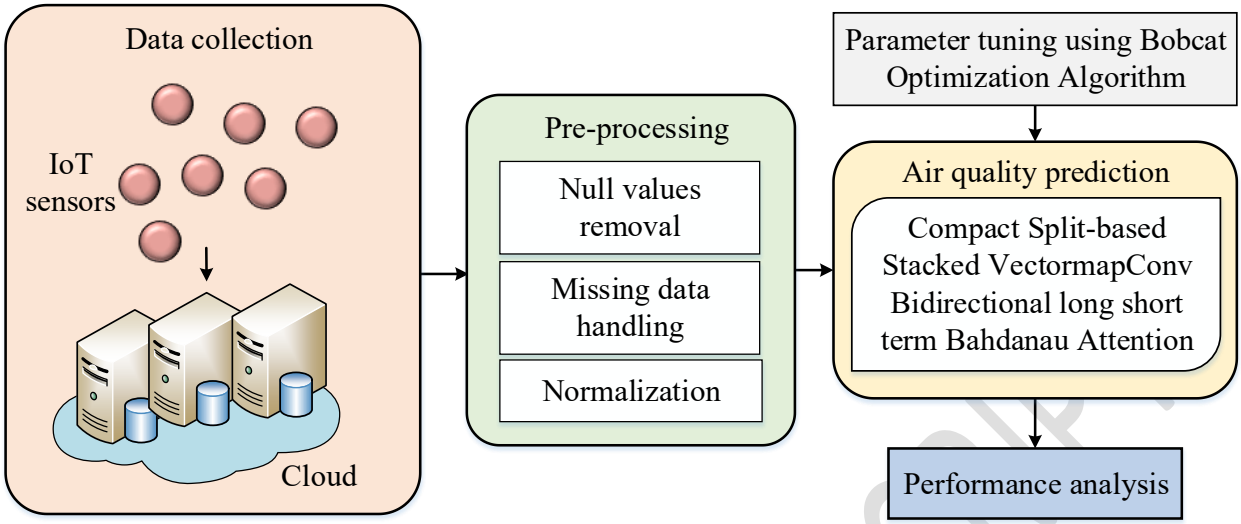


Figure 1: Block diagram of IoT-CAPM-DL model

3.1 Pre-processing

Preprocessing is a significant process that help to reduce data noise, which eventually speed up processing and increases the applicability of deep learning algorithms. Null values and missing data are the two most problems in extracting data and monitoring applications by Wang et al. (2022). Null value removal, missing data handling, and normalization based on L2-standardization are among the several operations carried out in IoT-CAPM-DL model during the data preprocessing step. To maintain data integrity, the rows or columns with significant number of null values is removed. For missing values, the missing value imputation technique is used by Niu et al. (2021). The L2-standardization normalizes the dataset so that the sum of squares of all values will equal to one by Benmamoun et al. (2024). The below equation illustrates L2-standardization in which y specifies the dataset's feature values in Equ (1).

$$|| y ||_2 = (| y_1 |^2 + | y_2 |^2 + ... + | y_p |^2)^{1/2} \quad (1)$$

After pre-processing, the pre-processed data are used to extract the significant features and then performs prediction.

3.2 Feature extraction and air quality prediction

In IoT-CAPM-DL model, compact split-based stacked vectormapconv bidirectional long short term Bahdanau attention (CSplitStack-VBA) network is used to extract features and predict the air quality parameters. CSplitStack-VBA network combines the strength of compact split-attention (CSplitA), CNN with vector map convolution, and stacked bidirectional long short term network (BiLSTM), and Bahdanau attention for improving air quality parameter forecasting. Each component of CSplitStack-VBA network plays a significant role in processing spatial and temporal data corresponds to air quality and enable accurate prediction. Furthermore, the benefit of stacking integration in CSplitStack-VBA network is that it allows several base learners to be integrated and fully utilize their differences, providing more thorough information during the model training process. Stacking integration can improve the resilience and performance of a model while reducing variance when compared to a single model. The architecture of CSplitStack-VBA network is provided in Figure 2.

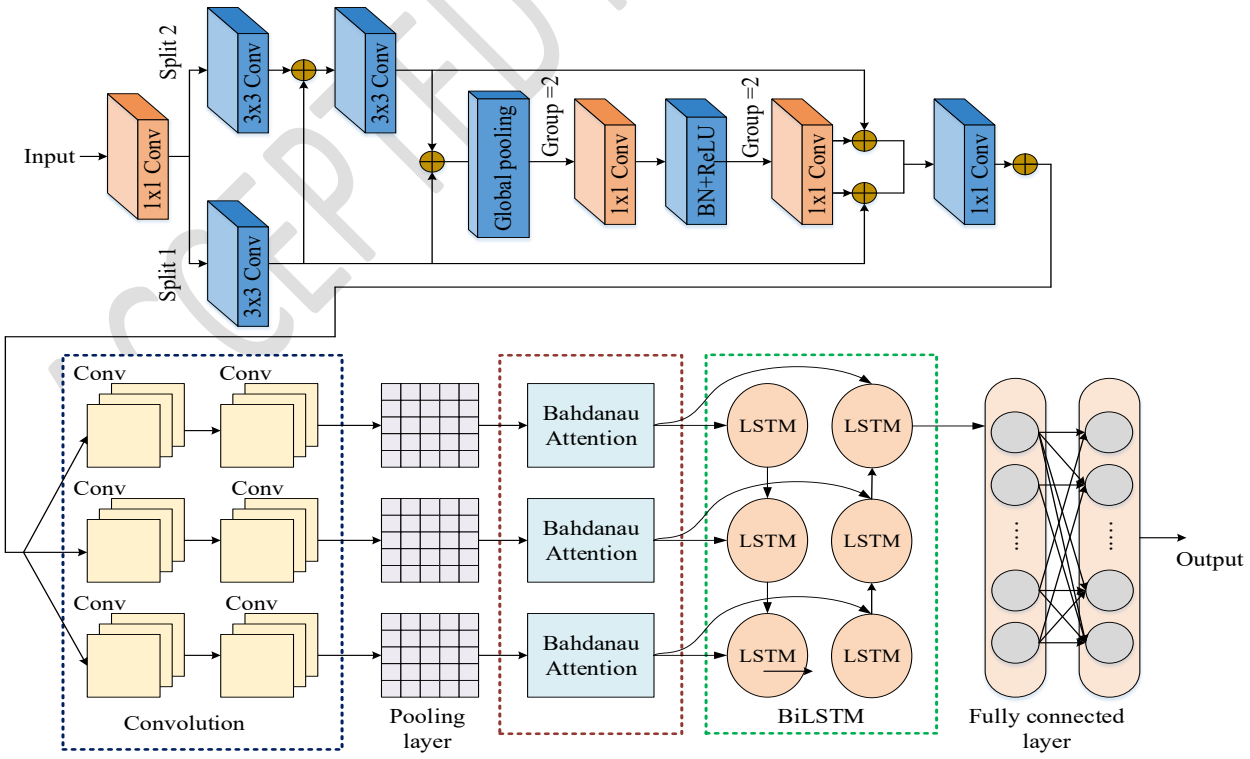


Figure 2: Architecture of CSplitStack-VBA network

3.2.1 Compact split attention module

The role of CSplitA model that improves the process of feature extraction by splitting feature maps into various groups and employing attention within each group has employed. The deep network architecture uses large channel shared groups for feature extraction. Moreover, this model utilizes the same convolutional processes with the same receptive field size for each group. In order to optimize the structure and enhance its applicability while reducing the overall number of parameters in the entire network, the CSplitA module has two feature groups ($Q = 2$). When these two groups are isolated from the input features, they undergo various transformation G_j. The two groups use a single 1×1 convolution followed by single 3×3 convolution. For increasing the representation across channels, the other group's (G_2) output feature maps can be subjected to additional convolution after adding the results of the first group (G_1). In this way, the network's reception area can be increased and information from both separated groups can be gathered. Therefore, a more robust capability to extract both local and global information from feature maps is presented by the CSplitA module. The following is a mathematical expression for the fusion feature maps in Equ (2):

$$\hat{V} = \sum_{j=1}^{W} G_j(Y_j), \quad \hat{V} \in \mathbf{R}^{J \times Y \times E} \tag{2}$$

where, J, Y and E resemble the output feature map scales. Channel-wise statistics designed by means of global average pooling are employed to gather the global spatial information. It is formed by compressing the results of the transformation across spatial dimensions, and d^{th} component through in Equ (3):

$$T_d = \frac{1}{J \times Y} \sum_{\beta=1}^{J} \sum_{\chi=1}^{Y} \hat{V}_d(\beta, \chi), \quad T \in \mathbf{R}^E \tag{3}$$

Channel-wise soft attention has employed to aggregate a weighted fusion characterized through a cardinal group representation since the split weighted combination can capture important data in feature maps. Furthermore, the feature map channel can be identified as follows in Equ (4):

$$W_d = \sum_{j=1}^{Q} b_j(d) G_j(Y_j) \tag{4}$$

where, b_j describes the (soft) assignment weight that can be expressed in Equ (5):

$$b_j(d) = \frac{\exp(\mathrm{M}_j^d(T))}{\sum_{j=1}^{Q} \exp(\mathrm{M}_k^d(T))} \tag{5}$$

where, M_j^d is measured by applying two 1×1 convolutions with ReLU activation and BatchNorm, it exemplifies the weight of global spatial information T to the d^{th} channel. As a result, the entire CSplitA model is simulated using an ordinary residual structure, and the output feature map is comparable to the input feature maps. Then, the result Z is calculated by the skip connection: $Z = W + Y$. Otherwise, the skip connection can undergo an additional transformation u. For example, u can be the convolution combined with stride or pooling and convolution.

3.2.2 CNN with vector map convolution

CNNs with vector map convolution offer a reliable method for managing and utilizing spatial and directional data, improving performance in applications that demand deep geometric and contextual knowledge. CNN is considered as a well-known deep neural network (DNN) that extracts complicated features and learns from the input data by employing convolutional operations in some of its layers rather than matrix operations. Typically, CNN encompasses three layers such as a convolutional layer, a pooling layer, and a fully-connected (FC) layer. After the network obtains the time-series data as inputs, the convolutional layer utilizes the pre-processed

data to extract complex features. Then, the result is fed to the activation function and the pooling layer. By using the average or max-pooling technique, the latter lowers the dimension of feature map. In order to upsurge learning stability and evade overfitting issue during training, batch normalization (BN) and dropout are generally added to the network. The convolution of the preceding input feature map y_j^{m-1} (the j^{th} input feature map of $(m-1)^{th}$ layer, and the convolution kernel x_{jk}^m which links j^{th} and k^{th} feature map yields the feature map y_k^m of the convolutional layer by applying a nonlinear activation function $g(.)$. The calculation process is exemplified by the following equation in Equ (6):

$$y_k^m = g\left(\sum_{j=1}^{N_k} y_j^{m-1} * x_{jk}^m + c_k^m\right) \tag{6}$$

Where, N_k indicates the number of inputs in k^{th} feature map, c_k^m resembles the kernel bias, and $(*)$ indicates the convolution operation.

3.2.3. Vector map convolution

Vector map convolution is a specific form of convolution employed in CNN. It involves employing convolution filters to vector data for extracting significant features in air quality prediction. Instead of scalar multiplication, the convolution encompasses vector arithmetic (such as dot products, norms). The weight sharing ratio in vector map convolution is represented as $\frac{1}{P}$, where P signifies the vectormap dimension Di_{vm}. Assume $Ve_{in}^3 = [v_1, v_2, v_3]$ be the input vector and $\omega^3 = [\omega_1, \omega_2, \omega_3]$ specifies the weight vector with $P = 3$. A permutation υ is performed on the inputs to make each vector multiplied through each weight vector element in Equ (7):

$$\upsilon(v_j) = \begin{cases} v_3 & j = 1 \\ v_{j-1} & j > 1 \end{cases} \tag{7}$$

A new vector, Ve^3 is formed by performing circularly right shifted permutation to Ve_{in}^3. The above equation can be utilized to find the weight $\upsilon(\omega^3)$ permutation. Thus, the output vector Ve_{out} is given in Equ (8):

$$Ve_{out}^3 = [\omega^3.Ve_{in}^3, \upsilon(\omega^3).Ve_{in}^3, \upsilon^2(\omega^3).Ve_{in}^3] \tag{8}$$

Where, "." indicates dot product. The elements of Ve_{in}^3 and ω^3 are combined linearly to provide the outputs Ve_{out}^3. Let $Ve_F = [A, B, C]$ be the weight filter matrix for the vectormap and $Ve_h = [X, Y, Z]$ be the input vector after linear combination. Then, the vectormap convolution between Ve_F and Ve_H for $Di_{vm} = 3$ in Equ (9):

$$\begin{bmatrix} R(Ve_F * Ve_h) \\ I(Ve_F * Ve_h) \\ J(Ve_F * Ve_h) \end{bmatrix} = LM \otimes \begin{bmatrix} A & B & C \\ C & A & B \\ B & C & A \end{bmatrix} * \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{9}$$

Where, LM represents the learnable matrix specified as matrix $LM \in \Re^{Di_{vm} \times Di_{vm}}$, which has been initialized by employing in Equ (10):

$$l_{j,k} = \begin{cases} 1 & j = 1 \\ 1 & j = k \\ 1 & k = Cal_j, \ where\, Cal_j = (j + (j-1)) \,\&\, Cal_j = Cal_j - Di_{vm} \ \ if\ Cal_j > Di_{vm} \\ -1 & else \end{cases} \tag{10}$$

Any dimensional hypercomplex convolution can be utilized by selecting Di_{vm} and allocating a new constant matrix $LM \in \Re^{Di_{vm} \times Di_{vm}}$ matching Di_{vm}. The mechanism used for vectormap weight initialization is comparable to the quaternion and complex weight initialization.

3.2.3 Bidirectional long short term network

BiLSTM builds an inverse LSTM layer on top of the long short-term neural network in order to process reverse time series. Its strong sequence modeling capabilities designates LSTM. It can effectively retain and transmit long-term dependency information while selectively forgetting

irrelevant data by applying memory units and gating mechanisms. This tackles the gradient vanishing and exploding issues that conventional RNNs encounter, permitting them to perform exceptionally well in challenging tasks like time series prediction and natural language processing. Among these, the following formulas are employed to determine the functions and gates inside LSTM neurons in Equ (11), Equ (12), Equ (13), Equ (14), Equ (15), Equ (16):

$$I_T = \rho(W_I \times [H_{T-1}, X_T] + B_I) \tag{11}$$

$$F_T = \rho(W_F \times [H_{T-1}, X_T] + B_F) \tag{12}$$

$$\widetilde{c}_T = \tanh(W_c \times [H_{T-1}, X_T] + B_c) \tag{13}$$

$$c_T = F_T \times c_{T-1} + I_T \times \widetilde{c}_T) \tag{14}$$

$$O_T = \rho(W_O \times [H_{T-1}, X_T] + B_O) \tag{15}$$

$$s_T = O_T \times \tanh(c_T) \tag{16}$$

Where, T indicates the input sequences, X_T resembles the input data of the present time step, H_{T-1} indicates the hidden state of preceding time step, W_I and B_I specify the input gate's weights and biases, and ρ states the Sigmoid activation function. Equation (11) demonstrates that the input gate is utilized for controlling the updation of new input information. Equation (12) characterizes the forgetting gate, which regulates whether the preceding time step memory is forgotten. The cell state updated through the input and forgetting gate is characterized by Equation (13), where $\widetilde{c}_T$ designates the current time step's candidate cell state. The cell state updated by means of the input gate, forgetting gate, and candidate cell state is characterized by Equation (14). The output gate that controls the outcome of the current time step's hidden state is exposed in Equation (15). Through the updated gate and the output cell state, Equation (16) displays how to compute the hidden state of present time step, where s_T indicates the hidden state of present time step.

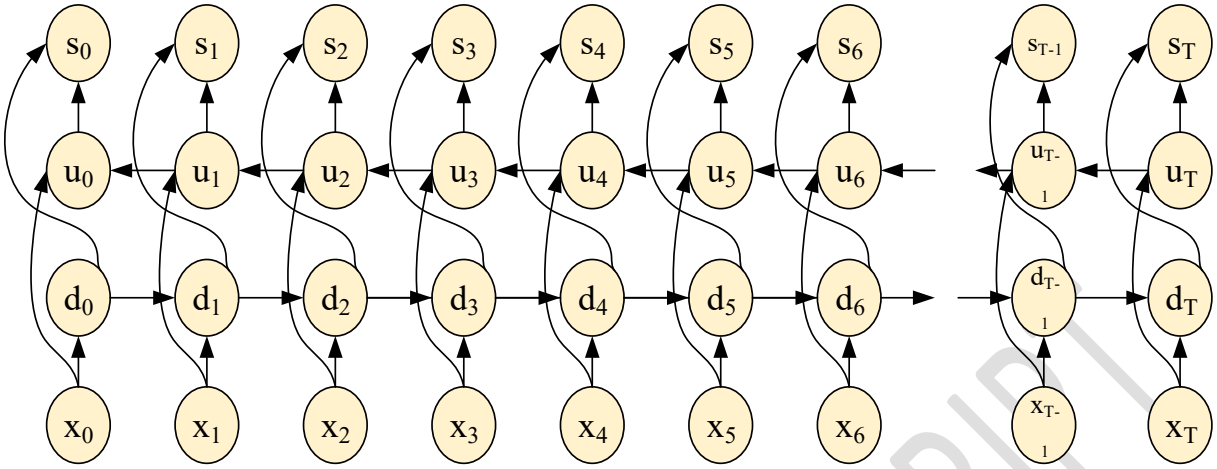


Figure 3: Structure of BiLSTM

Figure 3 depicts the structure of BiLSTM. Here, the input sequences of three successive time steps are employed. x indicates the input sequence set, such as the feature maps that vector map convolution and the attention mechanism have processed. A set of output sequences sent to the next FC layers is designated by s. The forward and reverse LSTM units are characterized by d and u. The bi-directional design of BiLSTM improves neural network performance and long-term temporal dependencies to produce more accurate prediction outcomes by strengthening its capacity to handle nonlinear time series. BILSTM has a computational efficiency issue since it necessitates bidirectional processing. By giving distinct weights to various features, the attention mechanism improves the perception and application of significant information by simulating how humans swiftly extract important information from huge amount of data, increasing processing efficiency and exactness of perceptual information. This makes more important features have a greater impact on the outcomes and minimize the computation complexity.

3.2.4 Bahdanau Attention

A technique that is often used in sequence-to-sequence models is Bahdanau Attention (BA), which provides distinct weights to various features. The encoder feature maps serve as keys k and values v, while the decoder LSTM generates a query vector Q_T at each time step T based on its present

hidden state. A scoring function is employed to calculate the attention scores, and the softmax function is then employed to normalize the results and acquire attention weights. The context vector, which delivers targeted information is computed as a weighted sum of the encoder feature maps and concatenated with the input of decoder LSTM in Equ (17).

$$Score(Q_T, K_I) = \tanh(w.(Q_T \oplus K_I) + B)) \tag{17}$$

Then, the attention weights β_T are attained by normalizing these scores by the softmax function in Equ (18):

$$\beta_T = soft\max(score(Q_T, k)) \tag{18}$$

By utilizing the attention weights, the context vector C_T is calculated as a weighted sum of encoder feature maps in Equ (19):

$$C_T = \sum_{I=1}^{P} \beta_{T,I}.V_I \tag{19}$$

Where, V_T indicates the I^{th} value vector from the encoder feature maps, and P resembles the number of feature maps. In order to provide the model with significant information during quality parameter prediction, the context vector C_T is concatenated with the decoder LSTM's input X_T *at each time step* T.

3.3. Hyper-parameter tuning

In IoT-CAPM-DL model, the bobcat optimization algorithm (BcOA) is employed in CSplitStack-VBA network to update the parameters for optimizing the loss function. The BcOA is one of the population-based optimizer that influences the members' search capacity to obtain appropriate solution for optimization problems in an iteration-based process. The design inspiration for BcOA resembles that the problem-solving space relates to the bobcats' wildlife habitat and their location within the habitat corresponds to the BcOA members' location in the problem-solving area. Consequently, in BcOA, the values for the decision variables are determined

by each bobcat (hyperparameters) as a population member based on the location it occupies in the issue solving space. Thus, each bobcat's location characterizes a potential solution to the problem, which can be mathematically described as a vector. Together, bobcats comprise the algorithm's population, which can be mathematically signified by a matrix in accordance with Equation (20).

$$Y = \begin{bmatrix} Y_1 \\ \vdots \\ Y_j \\ \vdots \\ Y_n \end{bmatrix}_{n\times m} = \begin{bmatrix} y_{1,1} \cdots y_{1,d} \cdots y_{1,m} \\ \vdots \ \ddots \ \vdots \ \iddots \ \vdots \\ y_{j,1} \cdots y_{j,d} \cdots y_{j,m} \\ \vdots \ \iddots \ \vdots \ \ddots \ \vdots \\ y_{n,1} \cdots y_{n,d} \cdots y_{n,m} \end{bmatrix}_{n\times m} \tag{20}$$

The primary bobcats' position is randomly initialized using the below equation in Equ (21),:

$$y_{j,d} = LB_d + s.(UB_d - LB_d) \tag{21}$$

where, P indicates the number of bobcats, s resembles a random number in the interval [0,1], m characterizes the number of decision variables, UB_d and LB_d specifies the upper and lower bounds of d^{th} decision variable, Y states the population matrix, Y_j signifies the j^{th} bobcat (candidate solution), and $y_{j,d}$ resembles its d^{th} dimension (decision variable). Equation (22) states that a vector can be utilized to signify the set of evaluated values for the fitness function in Equ (22).

$$G_j = Mini(LossFunction) \tag{22}$$

where, G_j specifies the computed fitness function depending on j^{th} bobcat. The best BcOA member resembles to the best assessed value for the fitness function, and the worst BcOA member resembles to the worst evaluated value for the fitness function.

The population members' locations are updated during the exploration phase using a simulation of bobcats tracking and moving behavior in the direction of prey during hunting. Equation (23) is

used to define the set of candidate prey for each bobcat, which is the location of other population members with a higher value for the fitness in Equ (23).

$$cp_j = \{Y_l : G_l < G_j \ and \ l \neq j\}, \text{ where } j = 1,2,...,P \text{ and } l \in \{1,2,...,P\} \quad (23)$$

where, Y_l represents the population member with higher fitness value than the j^{th} bobcat, G_l represents its fitness value, and cp_j specifies the set of potential prey positions for j^{th} bobcat. Using Equation (24), a new location is determined for every BcOA member. If this new position increases the fitness value based on Equation (25), it replaces the responding member's previous position in Equ (24) and Equ (25).

$$y_{j,k}^{Q1} = y_{j,k} + (1 - 2s_{j,k}).sp_{j,k} - J_{j,k}.y_{j,k}) \quad (24)$$

$$Y_j = \begin{cases} Y_j^{Q1}, & G_j^{Q1} \leq G_j \\ Y_j, & else \end{cases} \quad (25)$$

where, $s_{j,k}$ indicates the random numbers with interval [0,1], $J_{j,k}$ specifies the numbers randomly selected as 1 or 2, sp_j implies the chosen prey by j^{th} bobcat, $sp_{j,k}$ specifies its k^{th} dimension, $Y_{j,k}^{Q1}$ resembles the new location computed for the j^{th} bobcat based on exploration phase, $y_{j,k}^{Q1}$ indicates its k^{th} dimension, and G_j^{Q1} specifies its fitness function value.

During the exploitation phase, the population members' positions is updated based on the chasing behavior of bobcat. Equation (26) is used to define a new location for each BcOA member close to the hunting location depending on the modeling of bobcat's location change during the chasing process. Based on Equation (27), the corresponding member's previous position is replaced with this new one if the value of fitness raises.

$$y_{j,k}^{Q2} = y_{j,k} + \frac{1 - 2s_{j,k}}{1 + u}().y_{j,k} \quad (26)$$

$$Y_j = \begin{cases} Y_j^{Q2}, & G_j^{Q2} \le G_j \\ Y_j, & else \end{cases} \tag{27}$$

where, $Y_{j,k}^{Q2}$ resembles the new location computed for the j^{th} bobcat depending on exploitation phase, $y_{j,k}^{Q2}$ designates its k^{th} dimension, and G_j^{Q2} specifies its fitness function value. The pseudocode of BcOA for parameter tuning is provided in Algorithm 1.

Algorithm 1: Pseudocode of BcOA for parameter tuning

Start
Initialize the size of population P, maximum number of iteration U, fitness function and other variables
Create an initial population matrix randomly using Equation (21)
Compute the fitness function using Equation (22)
For $u = 1$ to U
  For $j = 1$ to P
Phase 1: Exploration phase (tracking and moving close to prey)
  Compute the set of prey for j^{th} member of BcOA using Equation (23)
  Compute new location of j^{th} BcOA member using Equation (24)
  Update j^{th} member of BcOA using Equation (25)
Phase 2: Exploitation phase (chasing to catch prey
  Compute new location of j^{th} BcOA member using Equation (26)
  Update j^{th} member of BcOA using Equation (27)
End
Choose the best candidate solution obtained so far
End
Output the best solution (optimal value of hyperparameters)
End

4. Results and discussion

The experimental outcomes of both IoT-CAPM-DL and prevailing techniques on the air quality dataset are discussed in this section. The University of Utah Air Pollution Monitoring Network dataset Salt Lake City provided the dataset, which was gathered between 2019-07-26 and 2021-

05-14. The python programming language has been used to implement the proposed IoT-CAPM-DL. The input of IoT-CAPM-DL comprises of time series window of preceding air quality measurements, which encompasses dissimilar air quality indicators like PM2.5, PM10, CO, SO2, O3 and NO2, meteorological data, temporal elements and geographical characteristics. The meteorological data includes humidity, temperature, air pressure, wind speed and direction. Time-related features encompass days, hours, months and seasonal trends. The output is the predicted value of air pollution index. The effectiveness of the IoT-CAPM-DL is compared with recently published research articles for air quality prediction. The hyperparameter tuning of IoT-CAPM-DL method is given in Table 1.

Table 1 Hyperparameter tuning of IoT-CAPM-DL method

Parameters	**Values**
No of epochs	100
Initial learning rate	0.01
Batch size	32
Maximum iterations	100
Activation Function	ReLU
Dropout rate	0.2
Optimizer	BcOA

4.1 Dataset description

This air quality dataset was created using 25 pollution sensors from Salt Lake City, Utah, USA's Air Pollution Monitoring Network, which are requested from the University of Utah's linked group [32]. Each air quality sensor provides a packet of data for 60 seconds (supposing that the monitor is operating ordinarily). Each pollution monitor has environmental sensors, like a temperature and humidity sensor (Texas Instruments HDC1080), an optical particle counter (Plantower PMS3003), and a sensor for identifying reducing and oxidizing gases (SGXSensorTechMiCS4514). Conferring to the device utilized to take the readings (one row per device per hour), the readings in this dataset are aggregated and averaged across an hour. The FEM

Tropospheric Ozone Equipment at the Hawthorne Monitoring Site, run by the Utah Department of Air Quality (DAQ), could provide the desired data. It is delivered for every 60 minutes. A DAQ system's hourly ozone values are attached to the relevant dataset row. Every one of the twenty-five air pollution sensors has more than twenty-five sets of ozone.

4.2 Performance indicators

The performance of the IoT-CAPM-DL method for air quality prediction is measured using the mean absolute error (MAE), mean absolute percentage error (MAPE), RMSE, and the coefficient of determination (R2-Score).

Mean ***absolute percentage error:*** MAPE is defined as a relative statistic that expresses the average value of a relative error as a proportion of the true value. The expression of MAPE is given as follows in Equ (28):

$$MAPE = \frac{1}{p}\sum_{j=0}^{u-1}\frac{Qj - Bj}{Qj}\Big|*100 \tag{28}$$

Where, p resembles the total number of data points or time steps, Qj specifies the expected value and Bj indicates the real value.

Mean absolute error: MAE determines the average magnitude of detection errors while disregarding their directions. It is deliberated as the average of the absolute differences between the actual and predicted values for every sample in the test set when every individual differences partake similar weight in Equ (29).

$$MAPE = \frac{1}{p}\sum_{j=0}^{u-1}\left|Qj - Bj\right|^2 \tag{29}$$

Root mean square error: The RMSE metric, which is measured by the standard deviation of the prediction errors, designates how far the data points are from the regression line. The prediction becomes more misaligned if the value is higher in Equ (30).

$$RMSE = \frac{\sqrt{\sum_{j=0}^{u-1} (Qj - Bj)^2}}{u} \quad (30)$$

Where, u indicates the total number of data points.

R2-Score: A metric known as the R2 Score is employed to assess how well a linear regression technique calculates variations in a dependent variable from variations in the independent variables.

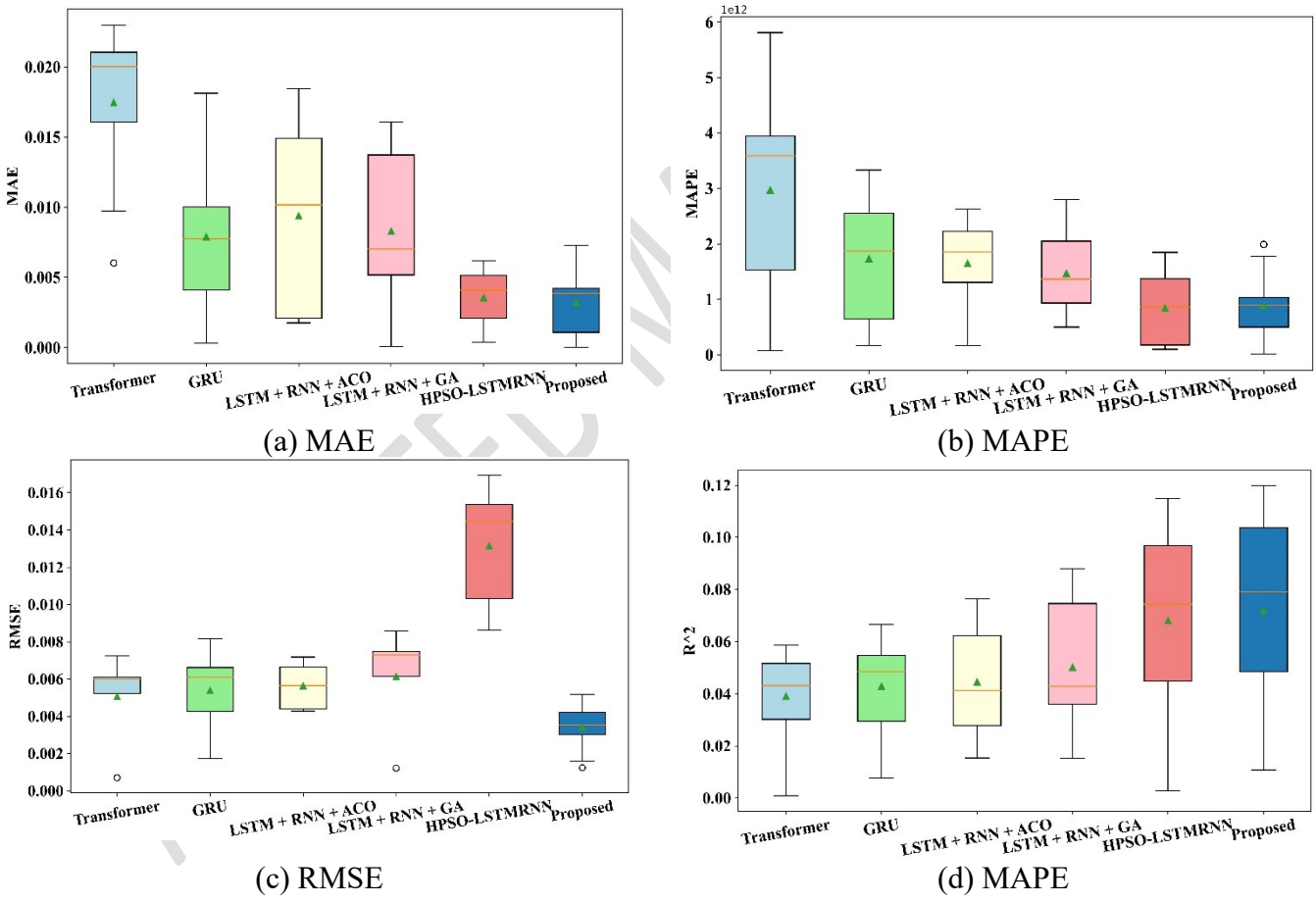
4.3 Performance comparison

In the experimental investigation, the IoT-CAPM-DL method has used the metrics RMSE, MAPE, MAE, and R2-Score to measure the performance of the IoT-CAPM-DL method. The result of IoT-CAPM-DL method is contrasted with the existing long short term memory (LSTM), support vector machine based regression (SVMR), gradient boosted tree regression (GBTR) and hybrid LSTM recurrent neural network (LSTM-RNN) models. In addition, other existing models are employed for comparison.

4.3.1 Comparison with different evaluation metrics

During the initial experiment, the IoT-CAPM-DL method is evaluated on PM2.5 on the employed dataset and computed the level of pollution in terms of RMSE, MAPE, MAE, and R2-Score. The comparative assessment of IoT-CAPM-DL method using different performance indicators like RMSE, MAPE, MAE, and R2-score is provided in Figure 4. The existing methods, including gated recurrent units (GRU), transformer, hybrid particle swarm optimization based HPSO-LSTMRNN, hybrid LSTM+RNN+genetic algorithm (GA), and hybrid LSTM+RNN+ant colony optimization (ACO) are used for comparison. The results show that the IoT-CAPM-DL model is significantly better, as demonstrated by its lower MAE and RMSE values, which signifies enhanced accuracy and less differences from actual data. The accurate predictions of model with a low percentage error are emphasized by the lower MAPE, which is crucial for dependable forecasting. The

maximum R2 score value designates that the IoT-CAPM-DL model is useful in capturing discrepancies in air quality and determines that it fits the data well. The CSplitStack-VBA approach, which incorporates BcOA for tuning parameters and the stacked neural network for catching temporal correlations, is responsible for the improved performance. This exemplifies the model's dependability and flexibility in predicting changes in air quality. The performance of the IoT-CAPM-DL method using different performance indicators is provided in Table 2.



(a) MAE (b) MAPE

(c) RMSE (d) MAPE

Figure 4: Comparative assessment of IoT-CAPM-DL method (a) RMSE, MAE, MAPE and R2-score

Table 2 Performance of IoT-CAPM-DL method using different performance indicators

Parameters	Transformer	GRU	Hybrid LSTM+RNN+ACO	Hybrid LSTM+RNN+GA	HPSO-LSTMRNN	Proposed
RMSE	0.0074	0.082	0.0089	0.0105	0.0184	0.0051
MAE	0.0237	0.0197	0.0185	0.0165	0.0082	0.0076
MAPE	$5874*10^9$	$3594*10^9$	$3021*10^9$	$2894*10^9$	$2002*10^9$	$1996*10^9$
R2-score	0.0591	0.0784	0.0874	0.0890	0.1227	0.1234

The effectiveness of IoT-CAPM-DL method in predicting air quality dynamics is demonstrated in Figure 5. The model has trained over 20 epochs, and the data is depending on information collected on the specified date. The graphical representation clearly illustrates the assessment of predicted and actual values, permitting for a visual assessment of the model's precision and ability to identify patterns in the air quality data. Besides, the model's ability to predict air quality is enhanced by utilizing 20 epochs, which implies the dataset has processed 20 times throughout training.

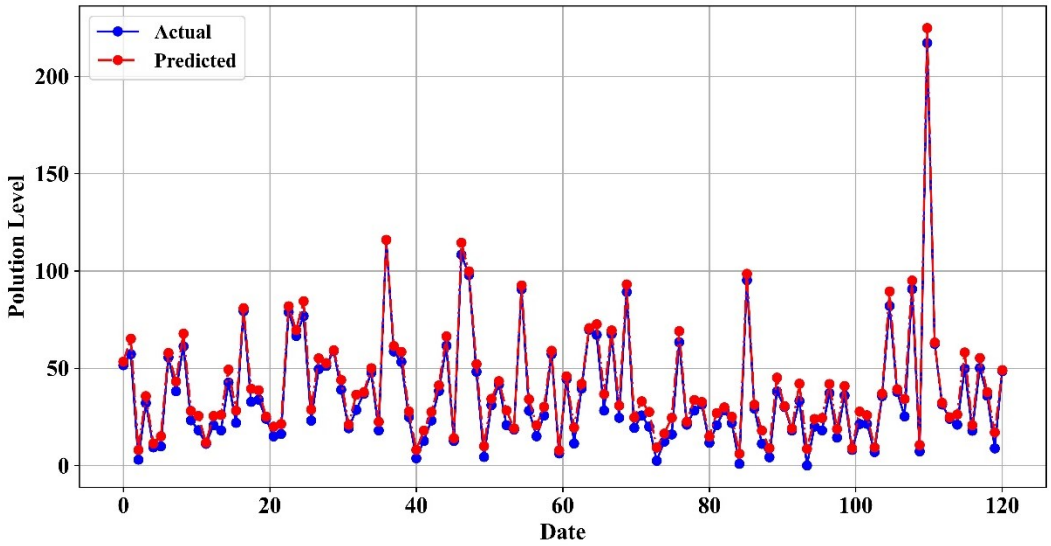


Figure 5: Prediction of IoT-CAPM-DL based on predicted and actual pollution level

The MAE results for IoT-CAPM-DL model and the existing models are displayed in Figure 6. The IoT-CAPM-DL model outperforms other existing methods such as LSTM, GBTR, SVMR and HPSO-LSTMRNN with MAE results of 1.92, 2.05, 2.53, 3.13, and 3.5 for 2 hours, 4 hours, 6 hours, 8 hours, and 10 hours. Among the existing methods, HPSO-LSTMRNN obtained MAE

value of 2.12 for 2 hours, 2.25 for 4 hours, 2.89 for 6 hours, 3.65 for 8 hours, and 4.12 for 10 hours, and it is closer to the proposed method. The IoT-CAPM-DL model performs better when the MAE value is lower.

The RMSE results for the IoT-CAPM-DL model and the prevailing models are displayed in Figure 7. In comparison to LSTM, GBTR, SVMR and HPSO-LSTMRNN models, the IoT-CAPM-DL model yields RMSE scores of 0.74, 0.87, 1.12, 1.46, and 2.30 for 2 hours, 4 hours, 6 hours, 8 hours, and 10 hours. Among the existing methods, HPSO-LSTMRNN obtained better RMSE values of 1.13 for 2 hours, 1.45 for 4 hours, 1.97 for 6 hours, 2.25 for 8 hours, and 2.87 for 10 hours and it is closer to IoT-CAPM-DL model. A lower RMSE of IoT-CAPM-DL model indicates that the model is performing better.

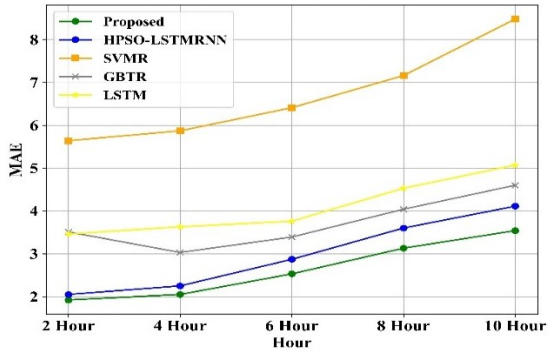


Figure 6: MAE comparison from 2 hour to 10 hour

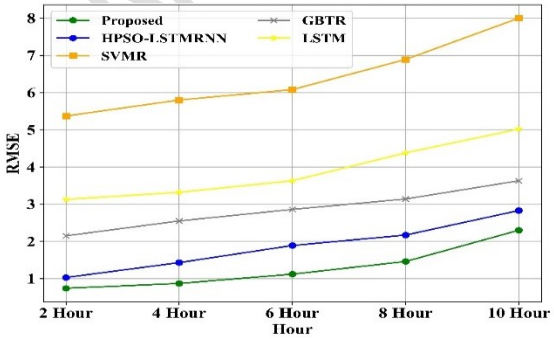


Figure 7: RMSE comparison from 2 hour to 10 hour

The MAPE outcomes of IoT-CAPM-DL model for the first two hours in the range of 15–30, 30–40, 40–70, and 70+ are displayed in Figure 8. According to "WHO" standard guidelines, PM 2.5 in the restricted range of 0 to 20 has less effect on the human body because of its lower value. In the same way, the MAPE outcomes for the first four hours in the range of 15–30, 30–40, 40–70, and 70+ are displayed in Figure 9. The graphical representation indicates that the IoT-CAPM-DL prediction approach performs better in terms of MAE, MAPE and RMSE errors as well as model expressiveness with various models.

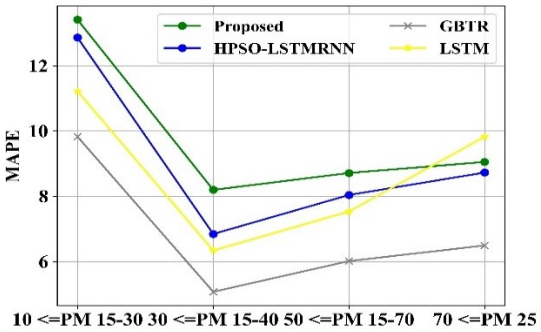


Figure 8: MAPE results for first 2 hour

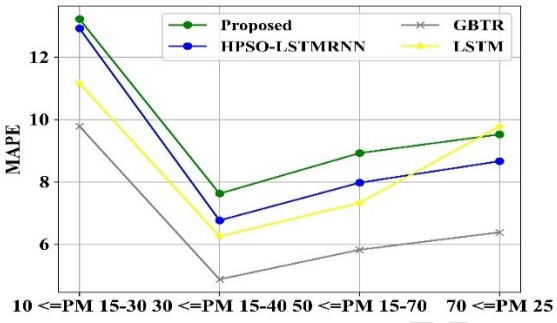


Figure 9: MAPE results for first 4 hour

4.3.2 Assessment for k-fold cross validation

The analysis of k-fold cross-validation is essential for examining an air-quality dataset. For effective public health management and environmental monitoring, air quality prediction depends heavily on the robustness and dependability of predictive models. The k-fold cross-validation is especially supportive to handle the complexity and inherent unpredictability of air quality data. Datasets on air quality usually reveal variations throughout time and space. This unpredictability can be helped and considered when training and evaluating models by including a range of data points from different sources. Besides, a model's generalizability across various contexts and time periods is evaluated by K-fold cross-validation through systematically rotating different subsets of the dataset for testing and training. Also, the k-fold cross-validation analysis is indispensable for getting a trustworthy and unbiased assessment of the model's accuracy, which increases the forecasts' accuracy for actual air quality conditions. When handling the complexities of fair quality datasets, the findings from K-fold cross-validation increase the comprehensive evaluation of the model's performance.

Table 3 Analysis of k-fold cross validation

Model	Fold	RMSE	MAE	MAPE	R2-score
	1	0.014	0.010	4.0%	0.93
	2	0.017	0.013	4.8%	0.89

Hybrid LSTM + RNN + GA	……	……	…..	…..	…..
	10	0.020	0.015	5.8%	0.84
	Avg.	0.018	0.013	5.2%	0.87
Hybrid LSTM + RNN + ACO	1	0.015	0.011	4.2%	0.92
	2	0.016	0.012	4.5%	0.91
	……	…….	……	….	….
	10	0.020	0.015	5.8%	0.84
	Avg.	0.018	0.013	5.2%	0.87
HPSO-LSTMRNN	1	0.012	0.009	3.5%	0.95
	2	0.014	0.010	3.8%	0.94
	……	…….	……	……	……
	10	0.015	0.011	4.0%	0.93
	Avg.	0.013	0.010	3.8%	0.94
Proposed	1	0.011	0.008	3.2%	0.91
	2	0.012	0.009	3.5%	0.93
	……	……	……	……	……
	10	0.014	0.010	3.7%	0.92
	Avg.	0.012	0.009	3.7%	0.93

The outcomes of a 10-fold cross-validation for IoT-CAPM-DL and existing models in air quality prediction are shown in Table 3. Every row resembles to a fold, presenting metrics like MAE, R2 Score, MAPE, and RMSE. The average performance over all folds is exposed by the "Avg" row. With an average RMSE of 0.012 and MAE of 0.009, the IoT-CAPM-DL model continuously establishes increased accuracy in predicting air quality levels and showing a decrease in prediction errors. The model's accuracy is emphasized by the average MAPE of 3.7%, which displays a lower level of error. A better level of explained variance and dependability in the predictions is designated by the R2 score of 0.93. The existing models show poorer performance with higher average higher RMSE, MAPE, and MAE, and lower R2 scores. The IoT-CAPM-DL model's dependability and effectiveness are emphasized by its strong and constant performance in air quality prediction.

5. Conclusion

This paper contributes to a novel IoT-CAPM-DL model for predicting air quality by addressing the existing problems. To ensure the data quality, the IoT-CAPM-DL starts with pre-processing the collected data stored in cloud. After pre-processing, the significant features are extracted and predicted air quality index using CSplitStack-VBA network. The optimal parameters selected through BcOA supported the CSplitStack-VBA network to minimize the error and maximize the performance. The performance of IoT-CAPM-DL model is evaluated by means of different performance indicators and it accomplished 1996*10^9 of MAPE, 0.0051 RMSE, 0.0076 MAE and 0.1234 R2-score, respectively. Overall, the proposed IoT-CAPM-DL model accomplishes better than the prevailing approaches across all performance indicators and contributes to a robust framework for predicting air quality and enhancing the understanding of air pollution dynamics to mitigate its effect on public health and environment. However, the continuously processing and transmitting IoT data can be resource intensive. This challenge is resolved by modelling a quality of service (QoS) aware and energy efficient protocol as future work. In addition, experimenting with other hybrid optimization strategies for hyper-parameter tuning can accomplish efficient neural network configuration.

Data Availability Statement: Dataset, (2023). https://ieee-dataport.org/documents/university-utah-airu-pollution-monitoring-network-salt-lakecity-ut-2019-07-26-2021-05-14

Conflicts of Interest: The authors declare no conflict of interest.

References

Ahmed, Abul Abrar Masrur, S. Janifer Jabin Jui, Ekta Sharma, Mohammad Hafez Ahmed, Nawin Raj, and Aditi Bose. "An advanced deep learning predictive model for air quality index

forecasting with remote satellite-derived hydro-climatological variables." Science of The Total Environment 906 (2024): 167234.

Babu, T. et al. (2024) “Integrated Early Flood Prediction using Sentinel-2 Imagery with VANET-MARL-based Deep Neural RNN”, Global NEST Journal, 26(10). Available at: https://doi.org/10.30955/gnj.06554.

Benmamoun, Zoubida, Khaoula Khlie, Gulnara Bektemyssova, Mohammad Dehghani, and Youness Gherabi. "Bobcat Optimization Algorithm: an effective bio-inspired metaheuristic algorithm for solving supply chain optimization problems." Scientific Reports 14, no. 1 (2024): 20099.

Dalal, Surjeet, Umesh Kumar Lilhore, Neetu Faujdar, Sarita Samiya, Vivek Jaglan, Roobaea Alroobaea, Momina Shaheen, and Faizan Ahmad. "Optimising air quality prediction in smart cities with hybrid particle swarm optimization-long-short term memory-recurrent neural network model." IET Smart Cities (2024).

Deepan, Subramanian, and Murugan Saravanan. "Air quality index prediction using seasonal autoregressive integrated moving average transductive long short-term memory." ETRI Journal (2024): e12658.

Duan, Jiahui, Yaping combination model optimized by dung beetle optimizer." Scientific Reports 13, no. 1 (2023): 12127.Gong, Jun Luo, and Zhiyao Zhao. "Air-quality prediction based on the ARIMA-CNN-LSTM

Feng, Tong, Yuechi Sun, Yating Shi, Jie Ma, Chunmei Feng, and Zhenni Chen. "Air pollution control policies and impacts: A review." Renewable and Sustainable Energy Reviews 191 (2024): 114071.

Idrees, Zeba, and Lirong Zheng. "Low cost air pollution monitoring systems: A review of protocols and enabling technologies." Journal of Industrial Information Integration 17 (2020): 100123.

Imam, Mohsin, Sufiyan Adam, Soumyabrata Dev, and Nashreen Nesa. "Air quality monitoring using statistical learning models for sustainable environment." Intelligent Systems with Applications 22 (2024): 200333.

Liao, Qi, Mingming Zhu, Lin Wu, Xiaole Pan, Xiao Tang, and Zifa Wang. "Deep learning for air quality forecasts: a review." Current Pollution Reports 6 (2020): 399-409.

Liu, Xian, Dawei Lu, Aiqian Zhang, Qian Liu, and Guibin Jiang. "Data-driven machine learning in environmental pollution: gains and problems." Environmental science & technology 56, no. 4 (2022): 2124-2133.

Liu, Bingchun, Xinpei Cao, Shiming Zhao, and Yan Xu. "Prediction and lag analysis of public concern about air pollution based on gray relation analysis and bidirectional long short-term memory." Air Quality, Atmosphere & Health 16, no. 5 (2023): 1037-1049.

Maio, S., G. Sarno, S. Tagliaferro, F. Pirona, I. Stanisci, S. Baldacci, and G. Viegi. "Outdoor air pollution and respiratory health." The International Journal of Tuberculosis and Lung Disease 27, no. 1 (2023): 7-12.

Malche, Timothy, Priti Maheshwary, and Rakesh Kumar. "Environmental monitoring system for smart city based on secure Internet of Things (IoT) architecture." Wireless Personal Communications 107, no. 4 (2019): 2143-2172.

Maltare, Nilesh N., and Safvan Vahora. "Air Quality Index prediction using machine learning for Ahmedabad city." Digital Chemical Engineering 7 (2023): 100093.

Nemade, Bhushankumar, and Deven Shah. "An IoT based efficient Air pollution prediction system using DLMNN classifier." Physics and Chemistry of the Earth, Parts A/B/C 128 (2022): 103242.

Niu, Zhaoyang, Guoqiang Zhong, and Hui Yu. "A review on the attention mechanism of deep learning." Neurocomputing 452 (2021): 48-62.

Prado-Rujas, Ignacio-Iker, Antonio García-Dopico, Emilio Serrano, M. Luisa Córdoba, and María S. Pérez. "A multivariable sensor-agnostic framework for spatio-temporal air quality forecasting based on Deep Learning." Engineering Applications of Artificial Intelligence 127 (2024): 107271.

Periasamy, S., Subramanian, P. and Surendran, R. (2024) "An intelligent air quality monitoring system using quality indicators and Transfer learning based Lightweight recurrent network with skip connection", Global NEST Journal, 26(5). Available at: https://doi.org/10.30955/gnj.006096.

Sachdeva, Shelly, Hitendra Singh, Shailee Bhatia, and Puneet Goswami. "An integrated framework for predicting air quality index using pollutant concentration and meteorological data." Multimedia Tools and Applications 83, no. 16 (2024): 46967-46996.

Saravanan, D., and K. Santhosh Kumar. "Improving air pollution detection accuracy and quality monitoring based on bidirectional RNN and the Internet of Things." Materials Today: Proceedings 81 (2023): 791-796.

Shaban, Wafaa Mohamed, Xie Dongxi, Kariman Samir Daef, and Khalid Elbaz. "Real-time early warning and the prediction of air pollutants for sustainable development in smart cities." Atmospheric Pollution Research 15, no. 7 (2024): 102162.

Shin, Sanghun, Keuntae Baek, and Hongyun So. "Rapid monitoring of indoor air quality for efficient HVAC systems using fully convolutional network deep learning model." Building and Environment 234 (2023): 110191.

Singh, Dharmendra, Meenakshi Dahiya, Rahul Kumar, and Chintan Nanda. "Sensors and systems for air quality assessment monitoring and management: A review." Journal of environmental management 289 (2021): 112510.

Sonawani, Shilpa, and Kailas Patil. "Air quality measurement, prediction and warning using transfer learning based IOT system for ambient assisted living." International Journal of Pervasive Computing and Communications 20, no. 1 (2024): 38-55.

Subramanian, S. et al. (2024) “An Automatic Data-Driven Long-term Rainfall Prediction using Humboldt Squid Optimized Convolutional Residual Attentive Gated Circulation Model in India”, Global NEST Journal, 26(10). Available at: https://doi.org/10.30955/gnj.06421.

Sundarapandi, A.M.S. et al. (2024) “A Light weighted Dense and Tree structured simple recurrent unit (LDTSRU) for flood prediction using meteorological variables”, Global NEST Journal, 26(8). Available at: https://doi.org/10.30955/gnj.06242.

Ullo, Silvia Liberata, and Ganesh Ram Sinha. "Advances in smart environment monitoring systems using IoT and sensors." Sensors 20, no. 11 (2020): 3113.

Venkatraman, M. et al. (2024) “Water quality prediction and classification using Attention based Deep Differential RecurFlowNet with Logistic Giant Armadillo Optimization”, Global NEST Journal [Preprint]. Available at: https://doi.org/10.30955/gnj.06799.

Wang, Xiaohu, Suo Zhang, Yi Chen, Longying He, Yongmei Ren, Zhen Zhang, Juan Li, and Shiqing Zhang. "Air quality forecasting using a spatiotemporal hybrid deep learning model based on VMD–GAT–BiLSTM." Scientific Reports 14, no. 1 (2024): 17841.

Wang, Lina, Xilin Deng, Peng Ge, Changming Dong, Brandon J. Bethel, Leqing Yang, and Jinyue Xia. "CNN-BiLSTM-attention model in forecasting wave height over South-East China Seas." Comput. Mater. Contin 73 (2022): 2151-2168.

Wu, Cui-lin, Rui-feng Song, Xing-hang Zhu, Zhong-ren Peng, Qing-yan Fu, and Jun Pan. "A hybrid deep learning model for regional O3 and NO2 concentrations prediction based on spatiotemporal dependencies in air quality monitoring network." Environmental Pollution 320 (2023): 121075.

Zhang, Zhen, Shiqing Zhang, Caimei Chen, and Jiwei Yuan. "A systematic survey of air quality prediction based on deep learning." Alexandria Engineering Journal 93 (2024): 128-141.