# CHAPTER 3

# IMPROVED SPARSE AUTO ENCODER USING DEEP LEARNING ALGORITHM

## 3.1 INTRODUCTION

Air pollution has emerged as a global crisis primarily due to the rapid expansion of industrial activities and the widespread use of automobiles, significantly impacting human health and environmental quality. The presence of particulate matter in polluted air allows tiny particles to enter the alveoli, hindering their function in gas exchange with the bloodstream. Prolonged exposure to these harmful substances can lead to lung cancer, heart disease, and other respiratory illnesses. In this study, a comprehensive approach is taken to enhance the prediction of air quality by leveraging neural network architectures such as Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and an Improved Sparse Auto-Encoder (ISAE). These models are designed to improve the understanding of spatiotemporal relationships in air quality prediction. By incorporating advanced machine learning techniques, the aim is to develop a robust predictive model capable of accurately forecasting air quality levels, thereby aiding in the mitigation of health risks associated with air pollution.

## 3.2 LONG SHORT TERM ARCHITECTURE

The Long Short-Term Memory (LSTM) is a neural network architecture introduced by Sepp Hochreiter and Jürgen Schmidhuber in 1997. It contains memory units, often called LSTM units, which are built from neural networks. Each LSTM unit includes several neural networks, featuring an input layer and an output layer. Typically, an LSTM unit comprises four fully connected neural network layers. These layers include three crucial gates: the forget gate, the input gate, and the output gate. These gates are essential for performing three primary memory management operations: deleting information from memory (the forget gate), adding new information to memory (the input gate), and using information stored in memory (the output gate). Additionally, the fourth neural network, known as the candidate memory, is used to generate new candidate information for possible insertion into the memory. The overall LSTM architecture and its components are depicted in Figure 3.1.

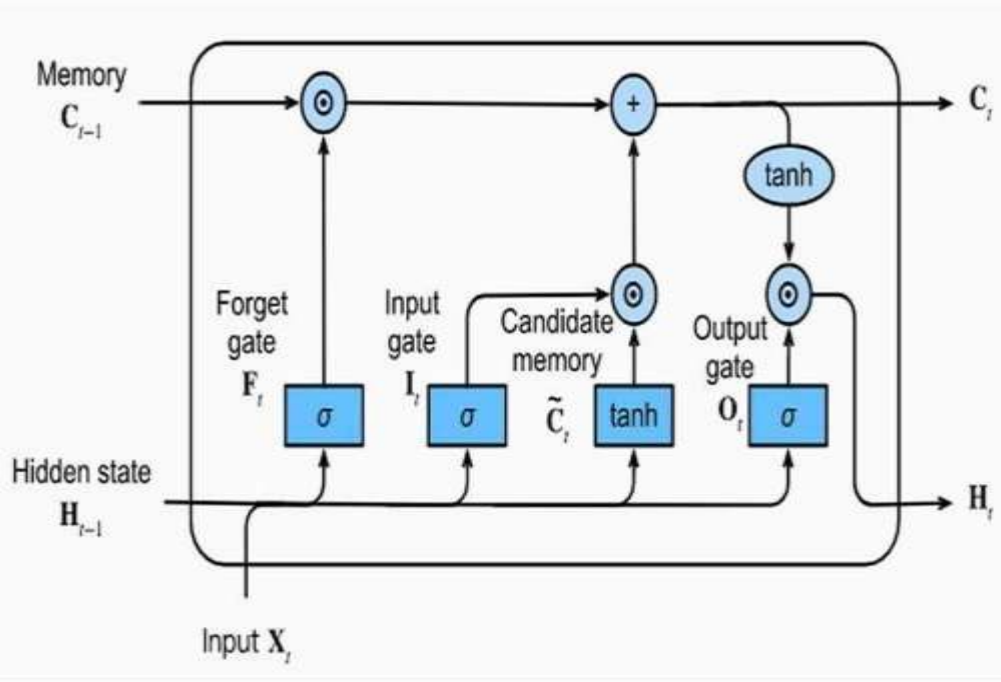


**Figure 3.1 Architecture of LSTM**

## 3.3 CONVOLUTIONAL NEURAL NETWORK

The prediction model is based on the Convolutional Neural Network (CNN), which compresses and extracts pertinent data using its convolutional and pooling layers. After passing through the CNN layer's output, the LSTM-based upper layer is in charge of generating predictions using the time series data. This forecast takes historical and present weather patterns and pollution levels into account. Initially, the time series data is converted into a set of matrices with predetermined dimensions. The features that are extracted from these matrices are then fed into the CNN. The output of CNN is used as the input of LSTM network. To arrive at the final prediction, the fully connected layer decodes the output generated by the LSTM network. This model shows the information flow and interaction between the CNN and LSTM layers to generate accurate predictions based on the input data. The predictive model of the Convolutional Neural Network is presented in Figure 3.2.

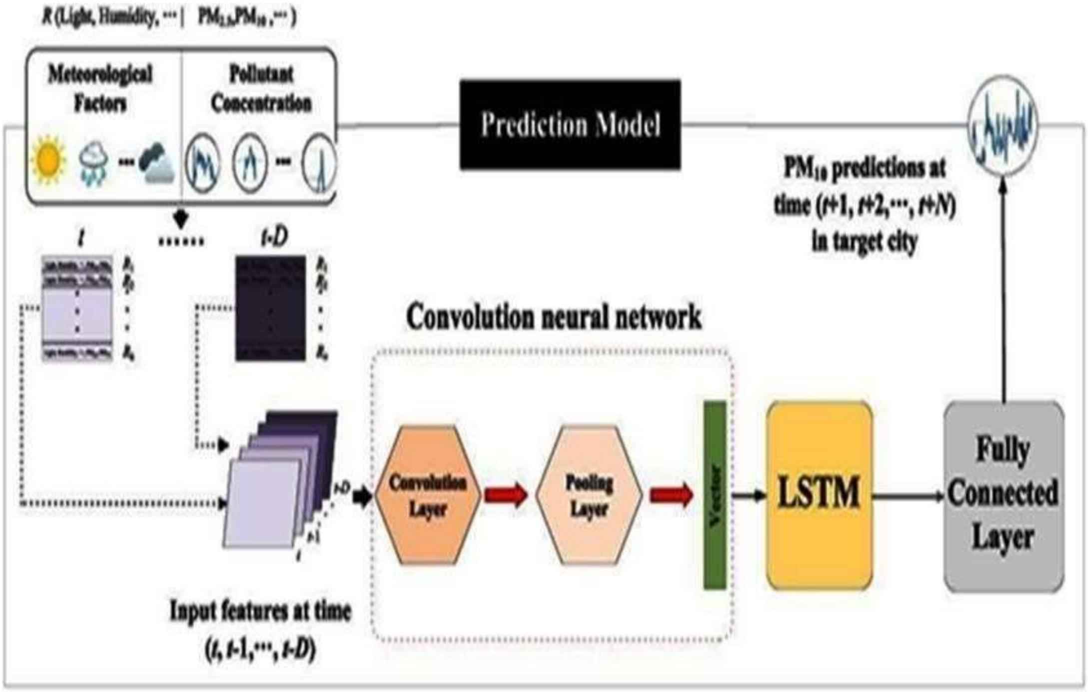


**Figure 3.2 Predictive model of Convolutional Neural Network**

**Limitations**

The time lag between measurements is crucial for making long-term predict the air quality prediction. Both LSTM and CNN are not suitable because of their respective training times and inability to be spatially invariant to the input data, respectively.

## 3.4 FEED FORWARD NEURAL NETWORK

A basic kind of artificial neural network is called a Feed Forward Neural Network (FFNN), in which data travels in a single direction from the input nodes to the output nodes via any hidden layers that may exist. The term "feedforward" refers to the absence of cycles or loops in this structure.

**Input Layer:** This initial layer comprises input nodes, with each node representing an input feature or attribute**.**

**Hidden Layers:** These intermediate layers lie between the input and output layers. Each hidden layer contains multiple neurons (nodes), with each neuron connected to every neuron in the preceding and succeeding layers.

**Output Layer:** This final layer produces ultimate output of the network. The number of nodes in the output layer depends on the specific problem being addressed.

**Connections (Weights):** Each connection linking neurons possesses an associated weight, dictating the strength of influence one neuron exerts on another. These weights typically undergo adjustment during training to enhance the performance of the network.

**Activation Functions: Neurons** typically process the weighted sum of their inputs through activation functions. Common examples include sigmoid, tanh, Relu (Rectified Linear Unit), and SoftMax (commonly used in the output layer for classification tasks).

**Bias Neurons:** Each layer (except the input layer) often features a bias neuron in addition to the input neurons. The bias neuron consistently maintains a value of 1 and aids the network in learning more complex functions.

## 3.5 AUTOENCODER

An autoencoder is a common unsupervised learning algorithm designed to reconstruct input data as accurately as possible. It typically consists of three distinct phases:

**Encoder:** The encoder phase involves a series of linear feed-forward filters, characterized by a weight matrix and bias parameters. Its purpose is to compress the input data into a lower-dimensional representation.

**Activation:** Following the encoding step, a nonlinear mapping, often referred to as an activation function, is applied to transform the encoded coefficients into a desired range, typically [0,1] or [-1,1].

**Decoder**: In the decoder phase, a series of reverse linear filters is utilized to reconstruct the input data from the compressed representation generated by the encoder. This reconstruction process is typically achieved using back-propagation, where the model's parameters are changed to minimize the error between the input and the reconstructed output.

## 3.6 SPARSE AUTOENCODER

A Sparse Autoencoder is a variation of the standard autoencoder that integrates sparsity into its architecture to create an information bottleneck. In particular, the loss function is designed to penalize activations within a layer, promoting a sparse representation of the data. Sparsity can be enforced through techniques such as L1 regularization or Kullback-Leibler (KL) divergence between the expected average neuron activation and an ideal distribution. The structure of a Sparse Autoencoder is depicted in Figure 3.3.

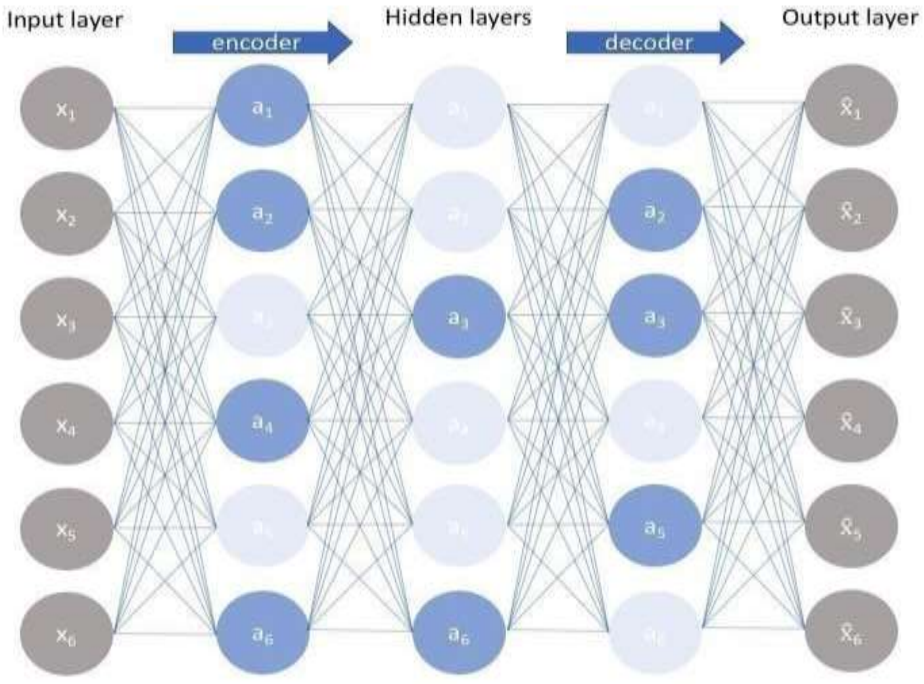


**Figure 3.3 Structure of Sparse Autoencoder**

- Input Layer: The input data is represented by this first layer, where each neuron is associated with a single input dataset feature.
- Hidden Layer: The input data compression takes place in the hidden layer. In comparison with the input layer, it has fewer neurons. In this layer, every neuron applies in an activation function, computes the weighted sum of the inputs from the previous layer, and produces an output. Only a tiny portion of neurons are active at any given time in a sparse autoencoder thanks to the activation function and sparsity constraint.

- Output Layer: The output layer makes an effort to reconstruct the input data using the compressed representation that the hidden layer taught it. In most cases, the number of neurons in this layer is the same as the total number of neurons.

## 3.7 PROPOSED METHODOLOGY

In the research work, an Improved Sparse Auto Encoder using Deep Learning based air quality prediction is proposed. This method is established to meet the necessities for flexibility and scalability for noise or anomaly detection in large systems Data with tens, hundreds, or even thousands of dimensions is well-suited for its use. While adding more samples does not change the calculation's complexity, it does increase training time because test data is handled correctly in increments rather than all at once during the training process. There are many different sampling techniques that the Improved Sparse Auto Encoder (ISAE) can be used. It describes how the suggested method, ISAE-DL, demonstrated better performance than the conventional deep learning techniques and neural network layer, such as ANN, LSTM, and CNN. Complex shapes can be derived from simpler shapes learned at lower network levels using the layered approach of the Improved Sparse Auto Encoder (ISAE) System.

The site which has the strongest spatiotemporal link to the target location is determined by utilizing features of the time series data that make up the spatial data. Next, these features are used to predict the target location. To determine which feature to retrieve for prediction, both time and space are considered. A collection of pertinent features = $\{f1$, f2, $f3$ …. $ltn\}$ and $Lt = \{lt1$, $lt2$, $lt3$ ….. $fn\}$ are considered. Equation (3.1) shows how longitude and latitude, which together make up a location's co-ordinate are measured from one point to another.

$$LtCi = (lti, pi, qi) \in Lt \qquad (3.1)$$

where $p_i$ and $q_i$ are represented in the latitude and longitude of the site respectively and the prediction process is enhanced by the relevant geographical factors. The equivalence of two sets of coordinates is the minimal difference between them and is given in equation (3.2),

$$= distloca(LtCy, LtCr)$$

$$= ((lty, py, qy), (ltr, pr, qr)) \tag{3.2}$$

where $Ltc_y$ is location coordinate and $p_y$ and $q_y$ are the latitude and longitude of the site

When it comes to determining related locations, the Spatial relationship sequence (SRS) of air quality data is interchangeable and is given in equation (3.3).

$$SRS = \{Ds1{,}2, Ds1{,}3, \ldots\ldots, Dsn-1{,}n\}, \qquad Dsi{,}i = 0, 0 < i < n+1 \tag{3.3}$$

where n represents total number of locations, 1 represents number of elements on the diagonal, and m represents an estimate of the matrix's members based on their coordinates $Dsi,\ I$ assigned as zero. The Feature Sequence Interval (FSI) of an area is calculated using the equation (3.4)

$$(l_{ti}, ft_i, t_{str.fini}) = \{(lti, fti, tstr), (lti, fti, tstr+1), \ldots\ldots\ldots, e(lti, fti, tfini)\},\ lti \in Lt, ft_i \in F, str < fini \tag{3.4}$$

In the case that has location and has features that varies between $str$ and $fini$, the time constraint is $str < fini$ and $(l_{ti},\ ft_i,\ t_{str})$ represents the ratio of $ft_i$ to $t_k$ as a measurement. Feature sequences is the distance between two locations of interest is given in equation (3.5).

$$Dt_{y,,.fini} = dist_{seq}\,(S(l_y, f_{tar}, t_{str.fini}), S(l_r, f_{tar}, t_{str.fini})),\ l_y, l_r \in Lty_{\neq r} \tag{3.5}$$

The equation (3.5) is used to gather information about the most relevant places in time.

where $f_{tar}$ for predicting the sequence of interest and any other parameters useful for predicting the air quality system. The spatial-temporal connection established is given in equation (3.6).

$$(lti,k) = (lti,) \cup (lti,), \qquad lt_i \in L \tag{3.6}$$

There is a risk that during intersection, candidate items will lose important target properties and the spatial temporal sequence,

$$(STRScandi(lt_r,k))[tlbc, tr] = (lti, ftar, t'str.fini, ) < tr < str' < fini' \tag{3.7}$$

The equation (3.7) is used to construct a model $M$ to represent the desired feature sequence, and return the desired feature sequence as a set, whereas s returns the desired feature sequence throughout the time interval $t'_{str}$ to $t'_{fini}$. The most relevant time series is generated from S and it is with to $t_r$, where the $lbc$ is a period of reflection. Figure 3.4 depicts the Proposed model of ISAE-DL

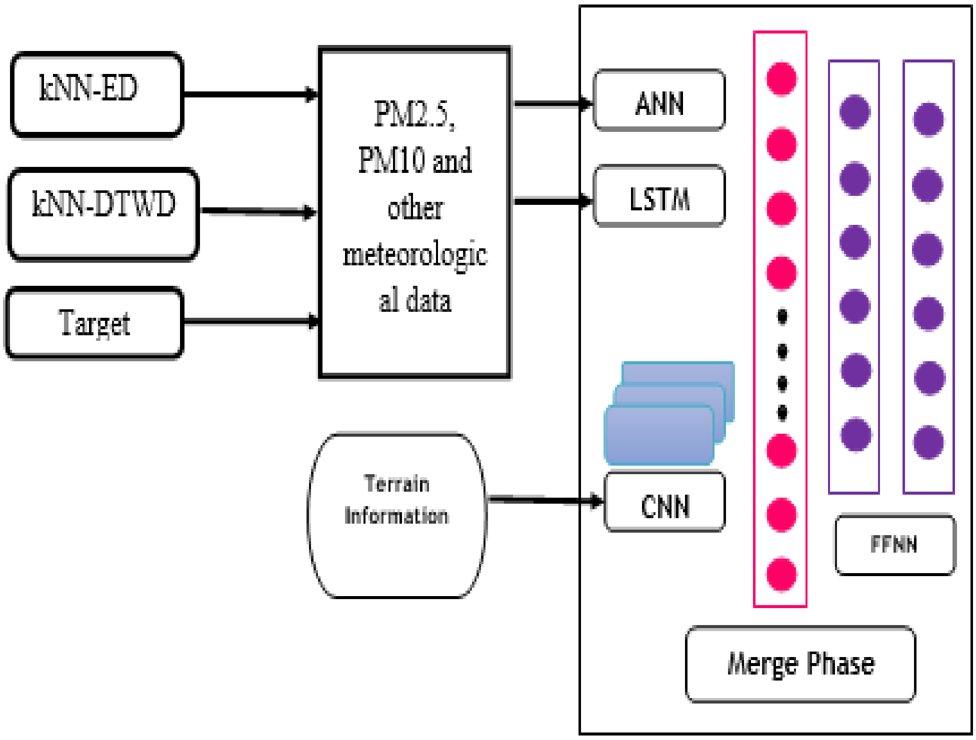


**Figure 3.4 Proposed Model of ISAE-DL**

The first stage involves training an improved sparse autoencoder (SAE), an unsupervised neural network, to learn the best representation of the training data. The second stage involves using an Artificial Neural Network (ANN), Long Short-Term Memory (LSTM), Convolutional Neural Network (CNN) to predict the air quality. The relevant locations are created from the training dataset .The associated and target location is calculated using spatial and temporal data. Training a predictive model is accomplished by the kNN-ED algorithm, which uses geographic coordinates to determine the Euclidean distance between points.

The Dynamic Time warping Distance (DTWD) is used to calculate time series data about the distance between two sequences, which can be used to correct for common scaling and shifting errors. Correction of the distortion of the sequence due to the Euclidean distance is achieved through data transformation with weights (DTW). kNN-DTWD identifies the strong temporal time series by ranking their distances and selecting the target location sequence. Unfortunately, the lack of complete data prevents these methods from **estimating** the degree of similarity between the time series. This problem is addressed by the filter short method, which employs the shortest threshold of interval and the data are transformed into the Dynamic Time Warping Distance. At the end, the kNN-DTWD algorithm determines the potential predictor.

The time series data is the characteristic model of The Temporal Relationship Extractor (TRE) method. Metrological data and other concentration input data are consistent and unbroken, falling into the categories of trends (low frequency) and rapid changes (high frequency). Because LSTM models learns past time series data and it is TRE and LSTM learns local trends. The ANN is extremely sensitive to sudden changes in the data it currently uses. Hence, the LSTM achieves low-frequency data, whereas the ANN achieves high-frequency data. It has been gained from earlier investigations; the feature relevance is variable to air quality (Zheng et al. 2013).

Pollutant dispersal affects air quality and emission is greatly impacted by its surroundings. As a result, utilizing meteorological information and AQI readings, a novel approach is created to forecast the target location's air quality. The sensitivity is enhanced by introducing the ANN, SRE method, which takes into the impact of the neighborhood data. FFNN is used in the improved sparse auto encoder, which effectively processes data with noise.

In order to improve the location correlation, Terrain Extractor (TE) includes data about the terrain. Andrews et al. (2017) identified that the elevation of each site was normalized as a correlation between dust and topography could be established and is given in equation (3.8),

$$El_s = \frac{ele - ele_{st}}{ele_{st}} \tag{3.8}$$

where e is elevation and Els is the elevation of spatial data and the conversion to the appropriate altitude is given in equation (3.9)

$$\text{Els} = \frac{1}{eEls} \tag{3.9}$$

where Els is the increasing performance and the effect is attenuated higher in the atmosphere since the distribution is height-dependent. In the end, the data from the TE, SRE, and TRE is fed into an ISAE-based FFNN. The improved sparse Auto Encoder is able to accomplish both layer merging and anomaly detection. An incremental method is used to properly manage the input data.

Autoencoders help to create more complicated shapes by decomposing input data into their component parts. It is detective tool in the Autoencoders features and flexible to retrieve complex data. For learning, both continuous and discrete data types are combined. In this technique, the input data is multiplied by a predetermined weight before being fed into a neural network layer and trained with FFNN techniques to predict the air quality prediction. The activation function are the linear function and log-sigmoid or hard limit, hyperbolic tangent, and sometimes saturation as a bias value. The estimated total can be calculated as shown in equation (3.10),

$$B = (wp + a) \tag{3.10}$$

where wp is predefined weight and $a$ is perceptron layer

While performing an estimation, perceptron often chooses a generic function, with the most popular is given in equation (3.11),

$$(p) = \frac{1}{(1+e^{-x})} \tag{3.11}$$

The error between the estimated value and the assumed value in the output is minimized if the weight is precisely estimated using the training data. Perceptron units are stacked in multiple layers to accommodate their large number, with output of each layer feeding into the input of the layer below it. This multilayer network is capable of solving complicated linear separable classification problems.

The input data was passed using the initial weights. Accuracy was calculated based on the difference between the observed and predicted values. The technique is carried out in batches, with each test being iterated and updated. Weights are changed using the median gradient value, and the new weights are applied to each input before the data is categorized based on the results. Figure 3.5 depicts the Architecture of Sparse Auto encoder.

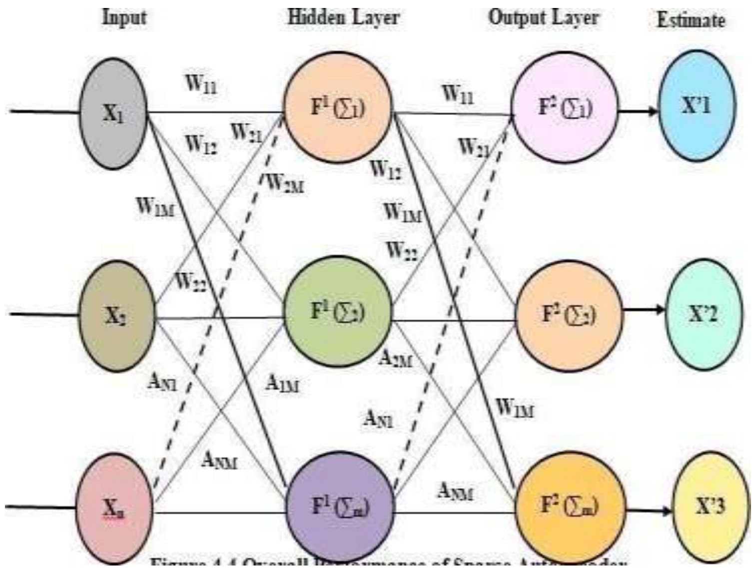


**Figure 3.5 Architecture of Sparse Autoencoder**

The newly acquired features from the improved sparse auto encoder deep learning algorithm was applied to neural network layer such as ANN,LSTM,CNN to create a more comprehensive system model that can be expanded upon by fusing additional data. In order to assess the shift in distributions, specific state data is used. It is possible to use both the continuous and discrete data for learning. The ISAE-DL is an efficient tool for detecting features. Building an ISAE denotes the perceptron, a neural network architecture that uses

Feed-Forward Neural network. When compared to the current ST-DNN method, the suggested system achieves effective prediction with higher accuracy.

**Limitations:** The intricate non-linear correlations between air pollutant concentrations and predictor factors are oversimplified by the Sparse Auto-Encoder (SAE) approach to air quality prediction. But it takes long time to train the system which collapse the performance efficiency. So, the improved version of this Autoencoder should be the better air quality prediction system.

## 3.8 EXPERIMENTAL RESULTS

The experimental results of the proposed method ISAE-DL are compared with other techniques ST-DNN, C-LSTME. The Air quality dataset was used for experimentation which was carried out using MATLAB R2016a. The Software specifications: Intel(R) CoreTMi3-4160T CPU@3.10 GHz 3.09 GHz processor, 4.00 GB RAM, Windows 8.1 pro, 64-bit OS, and 1 TB hard disc.

### 3.8.1 Dataset Description

The dataset for experimentation is used from the Kaggle repository (https://www.kaggle.com/rohanrao/air-quality-data-in-India). The data was Collected from Central Pollution Control board which is the official portal of the government of India(https://cpcb.nic.in/). The original dataset contains 53751 instances and 15 attributes

The dataset comprises hourly and daily measurements of the Air Quality Index and air quality data from multiple stations located in multiple Indian cities, gathered between 2015 and 2020. The cities are given below:

Ahmedabad, Aizawl, Amaravati, Amritsar, Bangalore, Bhopal, Brajrajnagar, Chandigarh, Chennai, Coimbatore, Delhi, Ernakulam, Gurugram, Guwahati, Hyderabad, Jaipur, Jorapokhar, Kochi, Kolkata, Lucknow, Mumbai, Patna, Shillong, Talcher, Tiruvananthapuram, and Visakhapatnam.

The attribute information is given below:

Date YYYY-MM-DD, City, PM2.5, PM10, NO, NO2, NOx, NH3, CO, SO2, O3,Benzene, Toluene, AQI, and AQI_Bucket. AQI_Bucket has six values such as good, satisfactory, moderate, poor, very poor, and severe.

The sample dataset of the city Delhi is given below:

| City | Datetime | PM2.5 | PM10 | NO | NO2 | NOx | NH3 | CO | SO2 | O3 | Benzene | Toluene | Xylene | AQI | AQI_Bucket |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Delhi | 02-01-2015 15:00 | 188 | 49.89 | 16.71 | 29.06 | 43.74 | 33.5 | 10.93 | 5.5 | 15.8 | 3.51 | 10.28 | 1 | 402 | Severe |
| Delhi | 02-01-2015 16:00 | 185.2 | 64.7 | 13.86 | 27.68 | 38.4 | 32.75 | 14.06 | 5 | 30.36 | 3.6 | 15.2 | 1.16 | 402 | Severe |
| Delhi | 02-01-2015 17:00 | 157.2 | 60.09 | 16.32 | 26.73 | 41.17 | 31.25 | 18.04 | 4.83 | 26.17 | 3.64 | 11.53 | 1.11 | 400 | Very Poor |
| Delhi | 02-01-2015 18:00 | 137.4 | 99.63 | 12.82 | 26.85 | 37.01 | 32.25 | 15.52 | 2.83 | 17.24 | 3.46 | 8.64 | 1.57 | 398 | Very Poor |
| Delhi | 02-01-2015 19:00 | 50.68 | 79.18 | 9.46 | 27.32 | 32.08 | 32 | 8.38 | 1.75 | 17.35 | 2.74 | 7.11 | 1.36 | 393 | Very Poor |
| Delhi | 03-01-2015 04:00 | 44.78 | 58.67 | 30.38 | 24.08 | 48.28 | 30.5 | 19.74 | 1.83 | 16.87 | 4.13 | 9.13 | 1.1 | 142 | Moderate |
| Delhi | 03-01-2015 05:00 | 51.44 | 77.92 | 23.6 | 23.73 | 41.25 | 31.25 | 12.27 | 2.5 | 16.88 | 3.39 | 8.63 | 1.33 | 103 | Moderate |
| Delhi | 03-01-2015 11:00 | 91.19 | 72.47 | 15.25 | 21.7 | 41.18 | 0 | 11.27 | 3 | 15.45 | 2.44 | 8.57 | 1.25 | 98 | Satisfactory |
| Delhi | 03-01-2015 12:00 | 87.84 | 72.14 | 12.58 | 22.94 | 39.58 | 0 | 9.59 | 3.25 | 15.53 | 2.36 | 5.54 | 1.25 | 90 | Satisfactory |
| Delhi | 04-01-2015 01:00 | 263.5 | 392.3 | 71.46 | 54.18 | 99.97 | 146.8 | 14.54 | 4.5 | 36.4 | 8.73 | 25.33 | 5.49 | 233 | Poor |
| Delhi | 04-01-2015 02:00 | 168.2 | 325 | 52.5 | 41.9 | 73.18 | 124.8 | 12.91 | 3.12 | 31.93 | 8.27 | 25.18 | 4.54 | 256 | Poor |
| Delhi | 29-06-2017 21:00 | 31.5 | 105 | 15.8 | 42.77 | 5.44 | 35.88 | 0 | 0 | 0 | 0.21 | 0 | 0 | 44 | Good |
| Delhi | 29-06-2017 22:00 | 35.5 | 90 | 17.1 | 33.15 | 4.72 | 34.53 | 0 | 0 | 0 | 0.14 | 0 | 0 | 43 | Good |

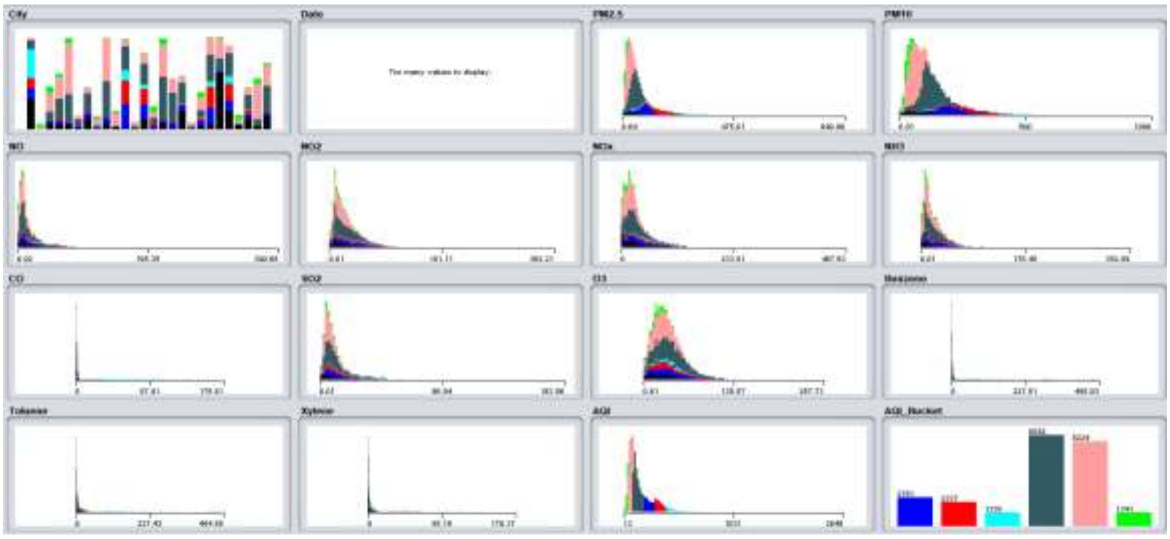

**Figure 3.6 Various level of Pollutant**

Figure 3.6 illustrates the fluctuating levels of various pollutants from the years 2015 to 2020, showcasing both increase and decrease in each pollutant over time. Notably,with the exception of CO, almost every pollutant depicted in the figure exhibits a seasonal trend. The primary focus of the plot is on the air quality index (AQI), which demonstrates seasonal variations. Additionally, there are gaps in the data for PM10, SO2, and O3 pollution between the years 2015 and 2020.

### 3.8.2 Data Preprocessing

The attribute 'xylene' was eliminated from the dataset since all of the values in this column were blank in each of the 26 cities. These 26 Indian cities have higher population densities than others, so the AQI (air quality index) and hourly and daily air quality data from various stations provide a good estimate of pollution levels. The datasets that were left over after each city's dataset was divided and cleaned had the following qualities:

- Ahmedabad: 2010 rows, 15 columns
- Aizawl: 102 rows, 15 columns
- Amaravati: 2148 rows, 15 columns
- Amritsar: 4295 rows, 15 columns
- Bangalore: 6304 rows, 15 columns
- Bhopal: 6593 rows, 15 columns
- Brajrajnagar: 7531 rows, 15 columns
- Chandigarh: 7835 rows, 15 columns
- Chennai: 9844 rows, 15 columns
- Coimbatore: 10230 rows, 15 columns
- Delhi: 12239 rows, 15 columns
- Ernakulam: 12401 rows, 15 columns
- Gurugram: 14080 rows, 15 columns
- Guwahati: 14582 rows, 15 columns
- Hyderabad: 16588 rows, 15 columns

- Jaipur: 16677 rows, 15 columns
- Jorapokhar: 18871 rows, 15 columns
- Kochi: 18896 rows, 15 columns
- Kolkata: 19847 rows, 15 columns
- Lucknow: 21856 rows, 15 columns
- Mumbai: 23865 rows, 15 columns
- Patna: 25723 rows, 15 columns
- Shillong: 26033 rows, 15 columns
- Talcher: 26958 rows, 15 columns
- Thiruvananthapuram: 28070 rows, 15 columns
- Visakhapatnam: 25932 rows, 15 columns

### 3.9 Performance Metrics Evaluation

The proposed method ISAE-DL is evaluated based on the Performance metrics. Accuracy, Precision, Specificity, Sensitivity, Area under curve, Mathew Co-relation Co-efficient. The performance of the algorithms is determined by the Confusion matrix which compares the predicated values and actual values. In the Confusion Matrix given in the Table 3.1, the columns denote the predicated values of a given class and the rows denote the actual values of a given class.

**Table 3.1 Confusion matrix**

| | | Actual Values | |
|---|---|---|---|
| | | Positive (1) | Negative (0) |
| Predicted Values | Positive (1) | TP | FP |
| | Negative (0) | FN | TN |

where TP is True positive, TN is True Negative, FP is False positive, FN is False Negative.

### 3.9.1 Accuracy

Accuracy is the measure to check the functionality of a classifier. The ratio of correct predictions to input samples determines the accuracy of a prediction. It is given by the equation (3.12),

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.12}$$

The accuracy value obtained by the proposed method, ISAE-DL for different iterations is given in Figure 3.7 and is compared with the existing methods, ST- DNN (Spatial Temporal-Deep Neural Network) and C-LSTME. (Convolutional-Long Short Term Memory Extended).

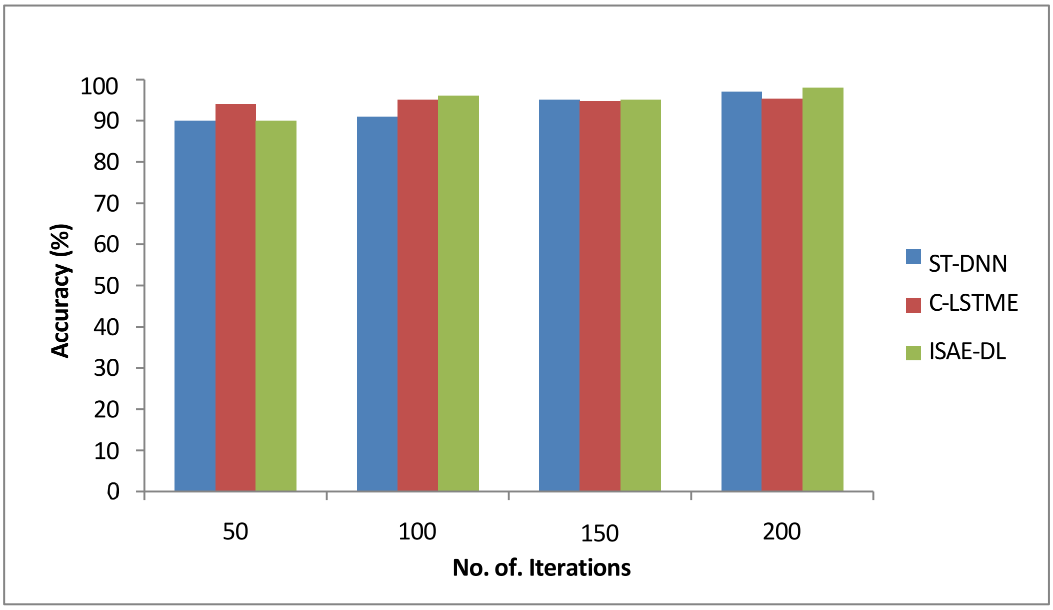


**Figure 3.7 Comparison of Accuracy**

From the Figure 3.7, it is evident that the accuracy of ST-DNN is 97%, 95.3% for C-LSTME and 98% for ISAE-DL. Prediction accuracy steadily improves with an increase in iterations. The accuracy of ISAE-DL has increased by 0.20% on comparing with ST- DNN and C-LSTME respectively which indicates that ISAE-DL exhibits higher accuracy value than ST-DNN and C-LSTME.

ST-DNNs handle spatial-temporal dependencies, they may struggle with complex long-term dependencies in time series data. This is often because standard DNNs lack the sequential memory mechanisms that recurrent models provide. The combination of convolutional and LSTM layers tends to slow down both training and inference, particularly when managing high-dimensional data. The Improved Sparse Autoencoder (ISAE) algorithm enhances traditional sparse autoencoders with adjustments that improve training stability, feature learning, and performance in high-dimensional data environments. Sparse autoencoders are known for learning meaningful feature representations by enforcing sparsity constraints, and improvements to this model have led to increased efficiency and robustness.

### 3.9.2 Precision

Precision is a degree of closeness of measurement of the True Positive value. It is given by equation (3.13),

$$Precision = TP/(TP + FP) \tag{3.13}$$

Figure 3.8 compares the precision of the proposed method, ISAE-DL with the existing methods, ST-DNN and C-LSTME for different iterations.

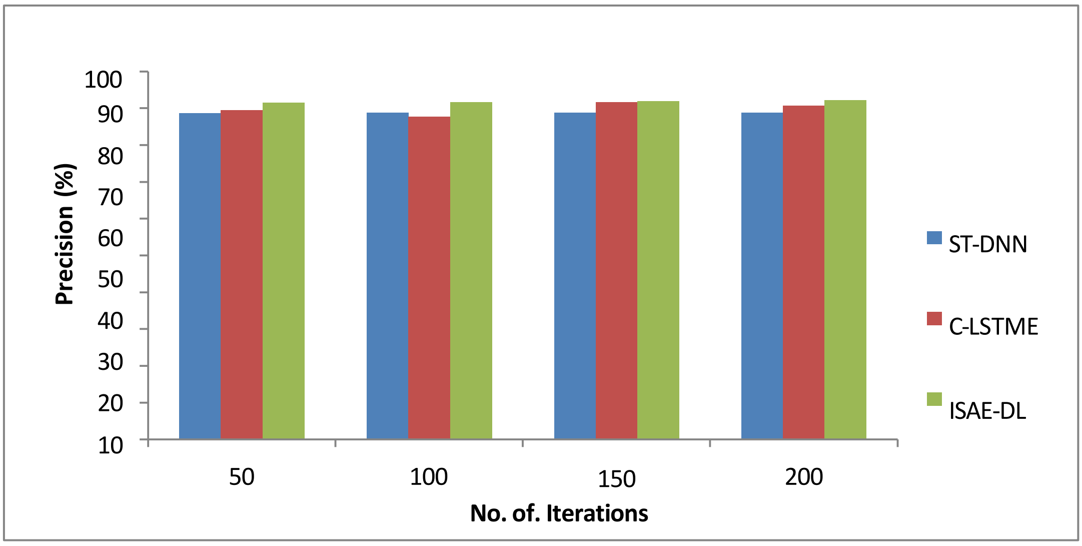


**Figure 3.8 Comparison of Precision**

The results indicate that the precision values of ST-SNN is 88% and C-LSTME is 90.7%, whereas, the proposed method ISAE-DL achieves 92.1%. The precision of ISAE-DL has improved by 1.4% and compared to ST-DNN and C-LSTME, which have lower precision values and ISAE-DL demonstrates higher precision values.

### 3.9.3 Specificity

Specificity is a measure of a model's ability to correctly predict negative values. It is given by the equation (3.14),

$$specificity = \frac{TN}{TN} + FP \qquad (3.14)$$

The Specificity of the proposed method ISAE-DL is compared with ST-DNN and C-LSTME for different iterations and is given in Figure 3.9

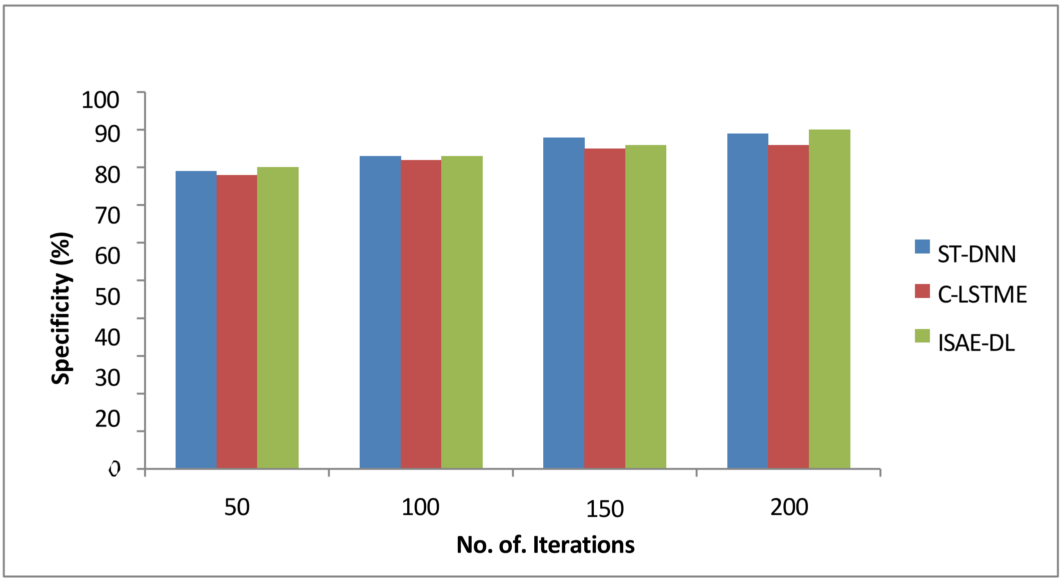


**Figure 3.9 Comparison of Specificity**

The experimental results given in Figure 3.9 show that the specificity of ST-DNN is 89%, C-LSTME is 86%, and the proposed method ISAE-DL is 90%. It is evident that the specificity of ISAE-DL has improved by 0.4% and compared to the existing methods, ST- DNN and C-LSTME. The results reveal that ISAE-DL demonstrates higher specificity values than ST-DNN and C-LSTME.

### 3.9.4 Sensitivity

Sensitivity is the ability of a model to predict true positives from each category. It is given by the equation (3.15)

$$\boldsymbol{sensitivity} = \frac{\boldsymbol{TP}}{\boldsymbol{TP}} + \boldsymbol{TN} \qquad (\mathbf{3.15})$$

The Sensitivity of the proposed method ISAE-DL is compared with ST-DNN and C-LSTME and is given in Figure 3.10.

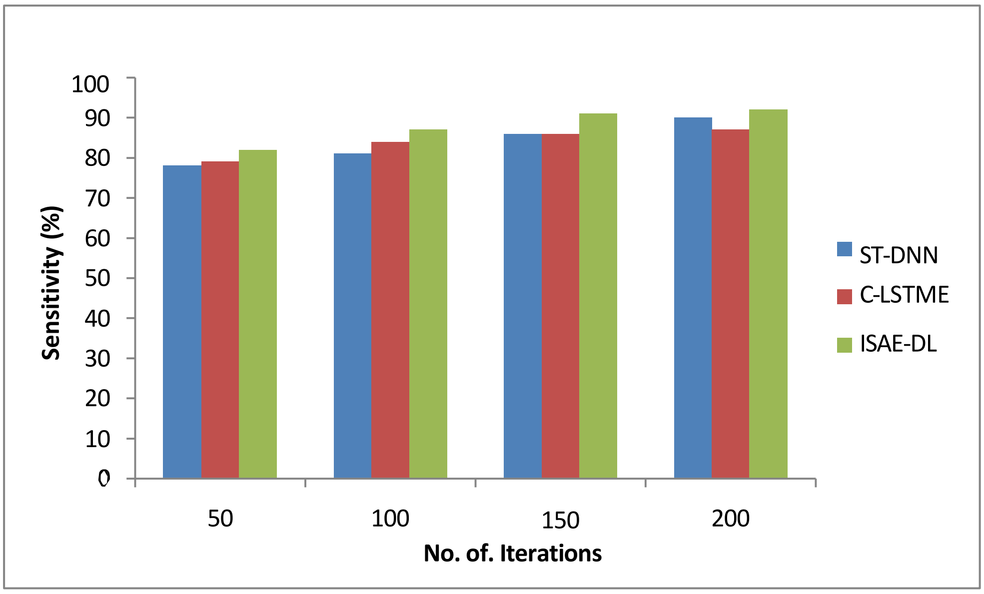


**Figure 3.10 Comparison of Specificity**

The experimental results of different iterations indicate that the sensitivity value obtained by ST-DNN is 90% and C-LSTME is 86%, while the proposed method ISAE-DL achieves 92%. The sensitivity of ISAE-DL has improved by 0.5% and 0.3% respectively. comparison, with ST-DNN and C-LSTME lower sensitivity values it shows that ISAE- DL demonstrates higher sensitivity values.

### 3.9.5 Area under Curve (AUC)

AUC is the most reliable classification indicator for skewed data since it does not depend on how the cases are distributed among classes. It is given by equation (3.16),

$$AUC = Sensitivity + \frac{Specificity}{2} \tag{3.16}$$

Figure 3.11 presents the comparison of Area Under Curve (AUC) values for the proposed ISAE-DL method with the existing methods, ST-DNN and C-LSTME.

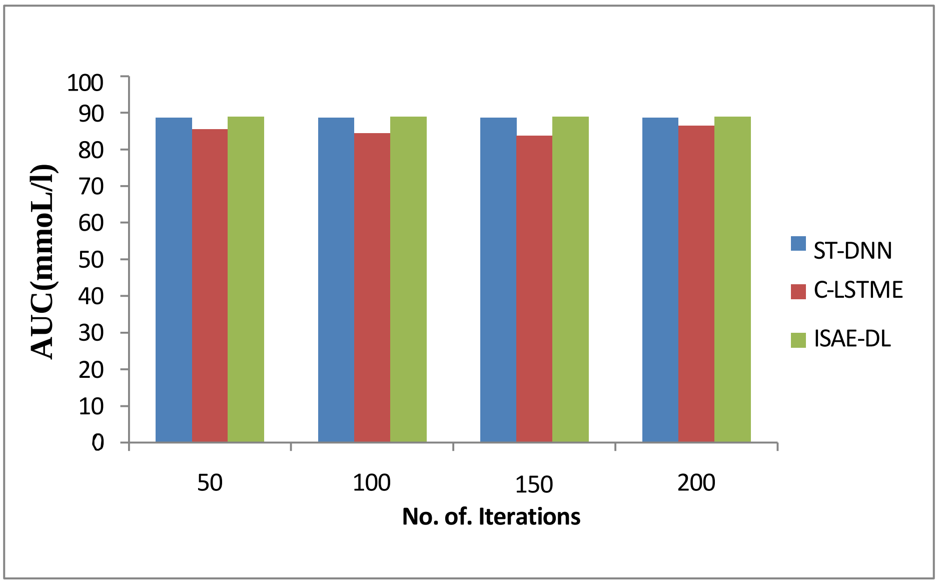


**Figure 3.11 Comparison of AUC**

Figure 3.11 illustrate that the AUC for C-LTSME is 86%, for ST-DNN is 88%, and the proposed method, ISAE-DL obtains 88%. The AUC of ISAE-DL has improved by 2.9% and 2.67% compared with C-LSTME and ST-DNN respectively. The existing models, ST- DNN and C-LSTME have low performance, as the models struggles to differentiate between the positive and negative classes. Whereas, ISAE-DL model is effective in distinguishing between the two classes and its predictions are reliable.

### 3.9.6 Matthew's Correlation Coefficient (MCC)

The MCC is a measure of the observed and predicated values in the air quality prediction system as given in equation (3.17),

$$MCC = TP * TN - FP * \frac{FN}{\sqrt{TP}} + FP)(TP + FN)(TN + FP)(TN + FN) \quad (3.17)$$

The Matthew's Correlation Coefficient of the proposed method ISAE-DL is compared with ST-DNN and C-LSTME for different iterations and is given in Figure 3.12.

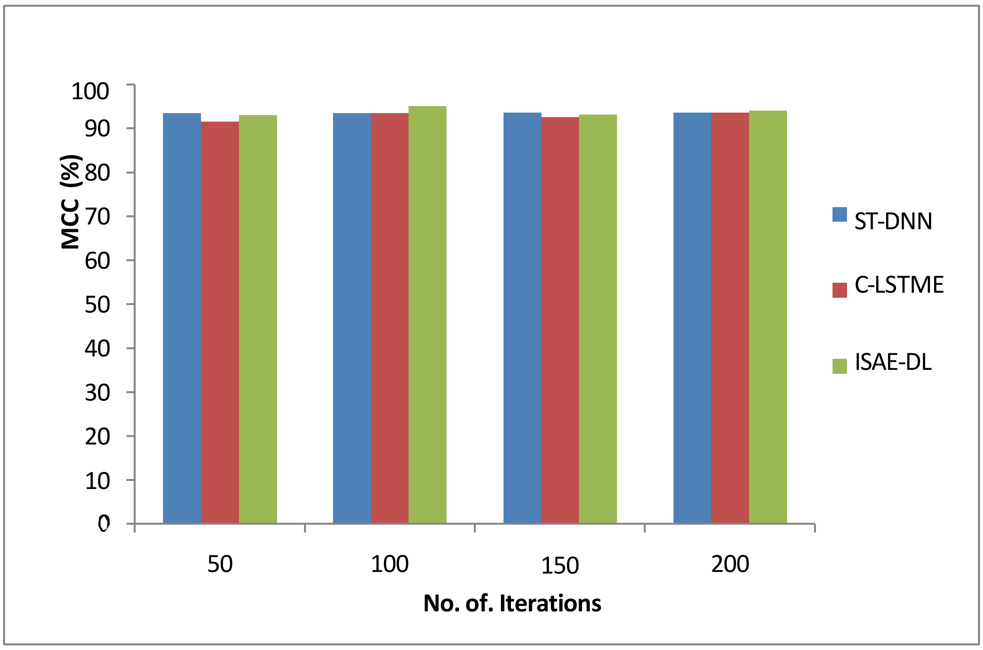


**Figure 3.12 Comparison of MCC**

The results of Figure 3.12 show that the ISAE-DL approach outperforms other existing techniques. The MCC of ISAE-DL has increased by 0.3% and 0.57% compared with ST-DNN and C-LSTME respectively. These results highlight how well the suggested ISAE-DL method performs in comparison to existing methods. Conversely, ISAE-DL attains higher MCC values, while ST-DNN and C-LSTME show lower MCC values.

### 3.9.7 - Mean Absolute Error Rate (MAER)

While converting data to digital format, errors can occur as a result of data transformation artifacts such noise, distortion, and interference. It measures the speed at which something is accomplished as a ratio. The error rate indicates how often a certain decision-making model assigns the wrong label to a given sequence. It is given in the equation (3.18),

$$MAER = FP + \frac{FN}{TP} + TN + FP + FN \qquad (3.18)$$

Figure 3.13 shows that the Comparison of Mean Absolute Error Rate (MAER) of ISAE-DL C-LSTME and ST-DNN.

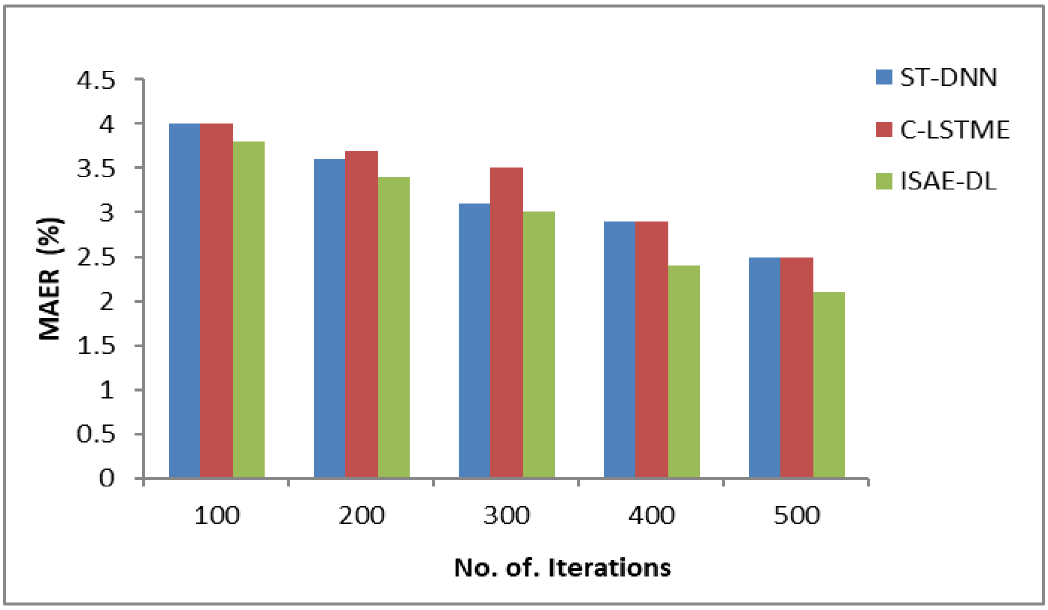


**Figure 3.13 Comparison of MAER**

This study reveals that with 500 training iterations, the error rate of ISAE-DL is 16% lower than that of the ST-DNN model and C-LSTME method. In comparisons ST-DNN and C-LSTME exhibit higher values, while ISAE-DL demonstrates lower values.

-

### 3.9.8 F-Measure

The F-measure (or F-score) is a statistical measure used to evaluate the performance of a classification model, particularly in binary classification problems. It is the harmonic mean of precision and recall and provides a single metric that balances the trade-off between these two.

$$Fmeasure = 2 * (precision * recall)/(\text{precision} + \text{recall}) \qquad (3.19)$$

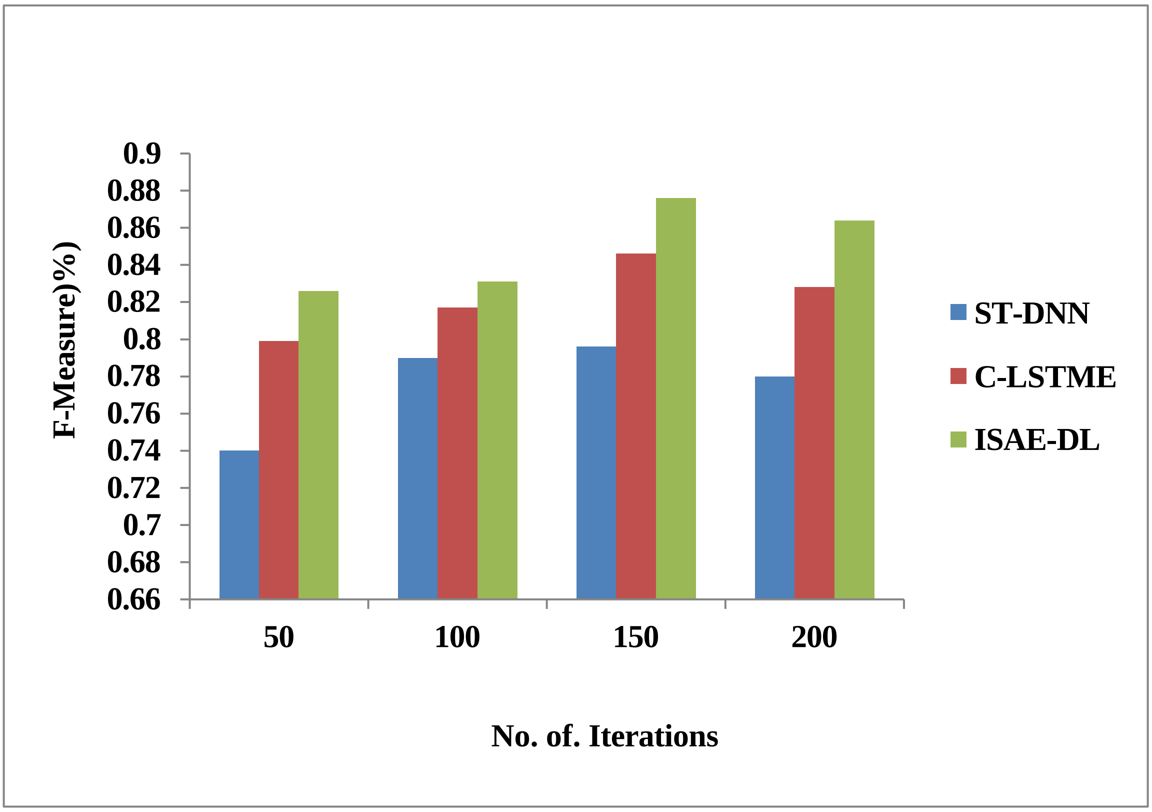


**Figure 3.14 Comparison of F-Measure**

The results of Figure 3.14 show that the ISAE-DL approach outperforms other existing techniques. The F-measure of ISAE-DL has increased by 4.35 % compared with C-LSTME respectively. These results highlight how well the suggested ISAE-DL method performs in comparison to existing methods. Conversely, ISAE-DL attains higher F-measure values, while C-LSTME show lower F-measure values.