IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Sentiment Analysis using Hybrid Model of Stacked Auto-Encoder based Feature Extraction and Long Short Term Memory based Classification Approach

Iqra Kanwal[1], Fazli Wahid[1], Sikandar Ali[1], Ateeq-ur-Rehman[1], Ahmed Alkhayyat[2], Akram AL-Radaei[3]
[1]Department of Information Technology, University of Haripur, Haripur 22620, Pakistan
[2]College of technical engineering, The Islamic University, Najaf, Iraq
[3]Information Technology Department, Thamar University, Yemen
iqrakhan9763@gmail.com, fazli.wahid@uoh.edu.pk, sikandar@uoh.edu.pk, ateeq@uoh.edu.pk, ahmedalkhayyat85@gmail.com
Correspondence should be addressed to Akram AL-Radaei (akram.alradaei@tu.edu.ye)

ABSTRACT The reviews of customer about a brand or product, movie reviews, and social media reviews, all can be analyzed through sentiment analysis. Sentiment analysis is used to identify the emotional tone of language in order to comprehend the attitudes, opinions, and feelings represented in online reviews. As for large data, it is a task that can take a lot of time and can be automated as the machine will learn through training and testing of data. Previously, various standard machine learning and deep learning models namely Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), Long Short Term Memory (LSTM), Naïve Bayes (NB), Support Vector Machine (SVM), Gated Recurrent Unit (GRU) have been used. The key issue on which our research is focused on is that when text is provided to LSTM directly, it cannot adequately extract informative features from the text, leading to less accurate findings. The Stacked Auto-encoder's softmax layer, when used directly to categorize the extracted features, is power-constrained and unable to do so accurately. A hybrid of the Stacked Auto-encoder and LSTM model has been proposed. SAE is used for relevant informative feature extraction. LSTM is used for further classification of sentiments based on extracted features. The proposed model has been evaluated on an IMDB dataset by splitting it in 5 different training testing ratios using performance evaluation metrics viz. confusion matrix, classification accuracy, precision, recall, sensitivity, specificity, and F1 score. The hybrid results perform best at a ratio of 90/10, and classifying sentiments with 87% accuracy. The proposed hybrid model's accuracy is better as compared to standard models namely RNN, CNN, LSTM, NB, SVM, and GRU.

INDEX TERMS Deep Learning, SAE, LSTM, Sentiment Analysis, IMDb, Classifier

I. INTRODUCTION

In recent years, sentiment analysis has emerged as a key topic of study in natural language processing due to its wide variety of academic, and industrial applications, and the rapid Web 2.0 growth [1]. In recent years, numerous techniques and tools for the specification of the document's polarity have been utilized. In majority of sentiment analysis applications [2], polarity detection is an essential binary classification task. For sentiment analysis, models trained on effective features with good design to get adequate results [3] of polarity classification were adopted in many earlier methods.

Standard classification methods viz. SVM, and NB was used by these models on linguistic elements including part-of-speech (POS) tags, n-grams, and lexical characteristics. This approach was having two drawbacks: (I) Feature space needed for model training is less and high-dimensional, lowering the performance of model; and (II) the feature engineering process is a task that takes a lot of time and effort. Several recent research studies [4, 5] have suggested and employed word embedding [6] to solve the drawbacks of traditional classification approaches stated above. A dense real-valued vector that takes into consideration numerous lexical associations is referred to as word embedding

[7, 8]. As a result, word embedding has become more prevalent as input for Deep Neural Network (DNN) in research of Natural Language Processing (NLP) [7]. Researchers in domains of computer vision [9], multimodal sentiment analysis [10], medical informatics [11], and finance [12] have been motivated by DNNs in the last few years.

The basic purpose of DNNs in textual data processing is to learn word embedding. Another goal is to use the learned feature vectors for conducting tasks of machine learning namely classification, and clustering [13]. The most often utilized deep networks in text processing research [13] are CNN and RNN. CNN and RNN learn local patterns and therefore popular in sequential modeling. RNN is helpful for a variety of text processing applications, but vanishing and exploding gradient problems arise in situations where the input data has long-term dependencies [7]. In many NLP applications, particularly sentiment analysis, these are the most prevalent dependencies.

To overcome this, LSTM and GRU networks were developed. Input, forget, and output gates are used in LSTM, whereas a reset gate and update gate are used in GRU. The standard RNN's difficulties can be overcome by utilizing LSTM and GRU. The issues faced by sequential models are handled by bidirectional LSTM (Bi-LSTM) and bidirectional GRU (BiGRU) in which hidden layer information flows in forward-backward direction. Bi-LSTM and Bi-GRU have two major flaws. The model would become highly complex due to the high dimension of input space and optimization will be difficult. Also, the critical contextual details of the text were ignored by the model. To address these challenges, CNN can be applicable for dimensionality reduction of feature space. CNN is also helpful for the extraction of text features [14]. All the encoded input vectors are combined into a weighted combination in attention based models [7], the largest weight will be assigned to the most relevant vectors. Two pre-trained word embedding; sentiment embedding and semantic embedding and LSTM were utilized [15] for the sentiment extraction and recognition of emotion. However, the model did not take into account the value of various sections of sentences. Bidirectional LSTM was merged with CNN [16] and the attention process was manipulated, but the issue of co-occurring short and long dependencies was not discussed. In [17], Rezaeinia et al. used CNNs to boost pre-trained word embedding, but long dependencies and terms of varying significance being ignored. The current research proposed a hybrid approach based on deep learning for polarity detection to fill the gap.

The main problem on which our research is carried out is that when we give text directly to LSTM, it cannot properly extract informative features from the text thus giving less accurate results. If we directly use the softmax layer of SAE for classifying the extracted features, the softmax layer cannot classify accurately because of its limited power.

The main focus of the study is a two-class problem that divides the text into positive and negative sentiments. Firstly, the main idea of our research is to use SAE to learn features and efficiently reduce the dimensionality of the features. LSTM took those features and is used for sentiment classification. We have used certain performance evaluation metrics which are accuracy, precision, sensitivity, specificity, confusion matrix, and F1 score. The main objective behind the research was to attain better testing accuracy by creating a hybrid of two models than standard deep learning models.

The contributions of the proposed study are elaborated in the following points:

1. A hybrid model comprising of SAE for feature extraction and LSTM as the classifier is proposed for sentiment analysis of movie reviews.
2. It will be helpful for a more accurate analysis of movie sentiments as positive and negative.
3. The accuracy of the sentiment analysis is improved up to 87% that is better than standard DL models.
4. It will minimize the efforts of people who are fond of watching movies and will be less time consuming to search for movie of their interest.

The work is structured as: Section II presents literature review. Section III demonstrates motivation behind the research. Section IV describes the proposed methodology. The experimental results are illustrated in Section V. The comparison of proposed hybrid model with k-fold cross validation is discussed in Section VI. Finally, conclusion and future directions are illustrated in Section VII.

II. LITERATURE REVIEW

Numerous models have been utilized for analyzing the sentiments. Deep learning models are in use as main classification module in sentiment analysis. In [18] a feed-forward neural network and Multilayer Perceptron (MLP) have been utilized for training. The accuracy using Feed Forward Neural Network (FFNN) is 77.45% and the accuracy using MLP is 67.45%. Feature selection is done by using a bag of words and n-grams features. The biggest disadvantage of utilizing n-gram features, particularly when n >= 3, results in the high-dimensional feature space. The intensity scores of emotion and sentiment are found using MLP classifier that is stacked on four individually trained models; CNN, LSTM, GRU, and SVR. The output of MLP will be the final intensity value. The proposed technique has been evaluated for sentiment analysis in financial domain. This ensemble model will improve the overall performance. Certain applications work better on Glove (Global Vectors for word representation) and others on Word2vec. Along with Glove and Word2vec, Auto encoder is utilized for better learning of word embedding in the prediction of the intensity of both emotion and sentiment. The accuracy for emotion analysis ranges from 74-77% and for sentiment analysis, it is 77-79% [19].

In [20], Neutrosophy and deep learning are combined for better sentiment classification as well as for effective sentiment prediction. The experiment is performed for sentiment analysis of tweets with BiLSTM using Glove, BERT (Bidirectional

Encoder Represntations from Transformers), RoBERTa (Robustly optimized BERT approach), MPNet and stacked ensemble models. Features are extracted using BiLSTM, GRU, Pre-trained Language Models, and stacked ensemble models, all of which are individually pre-trained. Feature Classification is trained in two ways; one by using two dense layers and the other through intermediate layers. Neutrosophy predicts the sentiment based on quantified sentiment in a sentence. The probability of each sentiment's prediction is calculated. The feature classification's final output while using SVNS Batch Norm calculated by intermediate layers is better as compared to the neural network's softmax layer [20]. Principal component analysis (PCA), Linear Discriminant Analysis (LDA), and Independent Component Analysis (ICA) have been utilized considerably for dimensionality reduction of features in sentiment classification. Machine Learning (ML) classifiers like SVM, NB, LR, and RF give less accuracy up to 79% and PCA is used for text feature dimensionality reduction but loss of information takes place during feature extraction [21]. LDA is used for data dimensionality reduction to eliminate the overfitting problem. LDA prefers to select a line that best divides the vectors but it has poor generalization performance and loss of information for sentiment analysis of tweets [22]. Text classifiers that use ICA to maximize the independent constituents of text documents and produce good classification results in many circumstances. Short-text documents, on the other hand, frequently contain less overlap in their feature terms, making ICA ineffective [23]. In recent studies, feature selection methods such as feature relation networks are also used to overcome this challenge [24].

The majority of recent DNN-based sentiment analysis research has focused on word embedding learning and afterwards, utilizing numerous DNN types for tasks involving clustering and classification. Words are represented via embeddings in n-dimensional vector space able to carry complex syntactic-semantic knowledge as well as encoding a wide range of linguistic patterns and regularities [25]. Experiments on the test set are conducted while using open source word vector representations i.e. GloVe [8]. It's a learning system that is unsupervised and developed by Stanford for word embedding generation from the corpus's global word–word co-occurrence matrix. The vocabulary words are mapped into fixed size dense embedding vectors by the embedding. For a better fitting with the neural network model, the embedding needs to be further trained. The embedding benefit is that they have a linear substructure, which means that related words in corresponding vector space would have similar Euclidean distances. They also save time and money by delivering features that are pre-trained for a variety of NLP tasks [26].

Words with similar meanings must have similar vectors. The primary assumption flaw is that frequently co-occurring vectors of semantically diverse words in confined neighborhoods are similar. The words comprising sentiments having opposite meanings can be represented by similar vectors as they often seem to be in similar contexts. Few scholars have offered sentiment-aware word vectors as a solution to this challenge. Large sentiment lexicons and supervised algorithms are used to construct these vectors [27] and thereafter, deep learning models have been used for the analysis of sentiments. In [28], the performance of three standard RNN structures; vanilla RNN, LSTM, and GRU is analyzed using pre-trained word vectors. For this purpose, three sentiment analysis datasets; movies reviews dataset: SST1, SST2 and Amazon health product reviews are used. For all the three datasets, the accuracy of Vanilla RNN ranges up to 75.7%, LSTM accuracy ranges up to 82.2% and GRU accuracy ranges up to 84.4%.

In [29], CNN is used as a classifier having 2 convolution layers and 3 pooling layers for sentiment analysis of movie reviews. Words with the same meanings are placed next to one another in vector space, which allows for the measurement and clustering of word similarity. The proposed model is compared with NB, SVM, and RNN. CNN performed better as compared to other models. Here the accuracy of CNN is nearly 45.5%. Traditional RNNs suffer from vanishing gradients, therefore LSTM is an enhanced RNN that overcomes this issue. As the RNN propagates backward in time, the gradients get smaller. As a result, moving data from early timestamps to later timestamps become more challenging [30]. LSTMs, on the other hand, solve this issue by having different gates and cell states. The cell state facilitates the transmission of relative information down sequences. It can be viewed as the memory of the network. Information would either be removed or added as the cell state descends the sequence, depending on the decision made by cell's gates.

LSTM with sentence representation (SR-LSTM) having two hidden layers is proposed. With a network of long short-term memory, the first layer learns continuous sentence vectors using pre-trained word embedding (Glove) for the representation of sentence. Sentence representation acts as an input to the second layer that will learn sentence relations which are encoded in document representation. Document representation is then used as a feature for sentiment analysis of movie reviews. The accuracy of SR-LSTM is 40-43%. The proposed model is compared with SVM, Naïve Bayes, RNN, LSTM, and GRU [31]. The attention mechanism is a very popular approach due to its low training time and parallel computation. Attention-based bi-directional LSTM is used for cross-language sentiment classification for a resource-rich language English and poor resource language Chinese. Source language English has labeled training data and target language Chinese has unlabeled data. The source language's labeled data and target language's unlabeled data are utilized in the training of the LSTM model and then sentiments are categorized in test data of the target language. The model is evaluated on book reviews, music reviews and DVD reviews. The accuracy of LSTM with and without sentence level and word level attention ranges between 81-82% [32]. A model for SA that combines LSTM with SVM is presented in [33].For evaluation, the IMDB movie reviews dataset was employed. Researchers tried to create a generalized sentiment analysis paradigm in [34]. In this

study, the authors' technique for producing vectors from the review dataset included CNN. The IMDB reviews were the source of the dataset for this investigation [34] .
SAE and SVM are utilized for classification of sentiments in IMDb movie reviews dataset. The SAE is used to input the features that were retrieved using continuous bag-of-words (CBOW). The SAE algorithm's hyper parameters are optimized with Genetic algorithm (GA). The SVM performs the final classification using the features that SAE has extracted. The check accuracy of SAE is 85.1% and SVM is 82.9% [35].

Transformers can only comprehend sequence dependencies by paying close attention. The input tokens are processed concurrently by transformers. Transformers are computationally efficient due to parallelization, but this also prevents the model from taking full advantage of the input's sequential nature [36]. Instead of utilizing the higher-level representations that are already available, the representation at particular layer is able to access representation from lower layers. Therefore, we suggested a model that might be shallow and compact but perform significantly better than Transformers of a similar size.

The Natural language processing with BERT for tokenization uses a total of 25000 comments as samples for a training data set, acting as the supervised learning approach [37]. It is a framework for improving performance at a high level that is based on the NLP algorithms used for the tasks of deep learning. This approach makes it simple to add and remove complexity. It manages massive amounts of data, needs little time for training, and uses little memory. The Novel BERT method has a greater accuracy of 83.5% when identifying the sentiment analysis of movies [37]. The BERT algorithm works in both directions to forecast the analysis on the dataset with the bidirectional encoder.

Many characteristics of an input text are captured by each layer of BERT. We look into the usefulness of features from various tiers. The model is then adjusted, and its test rate of test errors for several datasets, including IMDb, are recorded as being 4.21% [38]. BERT outperforms pre-training methods based on an autoregressive language modelling in terms of performance. A pre-train finetune disparity affects BERT. XLNet surpasses BERT in a variety of tasks, that included question answering, natural language inference, sentiment analysis, and document ranking, although XLNet accuracy for the IMDb dataset is only 85.4% [39].

In order to provide the audience with a reference for choosing movies, the article investigates the sentimental trend in movie reviews. The algorithm Term Frequency-Inverse Document Frequency (TF-IDF) is used to assess the significance of words in the reviews. The analysis of the movie reviews was accomplished using a SVM model, which achieved 85.2% accuracy [40]. Convolutional neural networks [41]are built to handle multidimensional data processing. The quantity of parameters that must be trained is decreased. The attention span may be beneficial as words in a phrase can much contribute to differentiating emotions. CNN is not utilized by the model only to extract local characteristics between texts but also employs bi-directional LSTM to collect semantic information globally for sentence context. The test accuracy rate is 86.132% [41].
The previously utilized approaches and enlisted models are depicted in Table I.

Table I. Comparison of various previous models in the literature with the proposed model showing better performance of the proposed one

S.No.	Author Name and Year	Methodology	Model	Dataset	Accuracy
1.	Akhtar et al., 2020 [19]	An ensemble model for emotion analysis and sentiment analysis. Denoising Auto-encoder is used for feature extraction. Multilayer Perceptron (MLP) stacked on four individually trained models; CNN, LSTM, GRU, Support Vector Regression (SVR)	MLP, SVR, CNN, LSTM, GRU	Microblog, News	77-79%
2.	Sharma et al., 2021 [20]	DL and Neutrosophy for quantification of each sentiment. Feature extraction using (i) Bi-LSTM-GRU with Glove embedding. Feature Classification layer training using (i) Dense layer, Softmax layer (ii) Intermediate Layers – output of feature extraction and batch normalization layer used for quantifying each sentiment and prediction	Bi-LSTM-GRU, Neutrosophy ,	SemEval 2017 Task 4	71%
3.	Baktha and Tripathy, 2017 [28]	three standard RNN structures are analyzed using pre-trained word vectors	RNN, LSTM, GRU	Amazon health product reviews, SST-1 and SST-2	RNN 75.7% LSTM 82.2% GRU 84.4%
4.	Dhola and Saradva, 2021 [42]	Comparative Analysis of ML and DL classifiers is performed on the Twitter dataset. ML Classifier Multinomial Naïve Bayes has less accuracy that is improved by hyper parameter tuning.	BERT, LSTM	Twitter	LSTM 80% BERT 85.4%

5.	Ouyang et al., 2015 [29]	Word2vec is utilized for feature extraction and CNN as a classifier	CNN	Movie reviews	45.5%
6.	Ramadhani and Goo 2017 [18]	Feedforward neural network and MLP are used for training	FFNN and MLP	-	67.45-77.45%
7.	Ahmed K. et al., 2022 [35]	The SAE hyper parameters are optimized with GA The SVM performs the final classification using the features that SAE has extracted.	SAE, SVM	IMDb	SAE 85.1% SVM 82.9%.
8.	CH Kumar and RS Kumar, 2022 [37]	The Bidirectional encoder representation for transformers manages massive amounts of data, needs little time for training, and uses little memory. BERT algorithm works in both directions to forecast the analysis on the dataset with the bidirectional encoder. It acts as the supervised learning approach.	BERT	IMDb	83.5%
9.	SichangSu, 2022 [40]	The algorithm Term Frequency-Inverse Document Frequency (TF-IDF) is used to assess the significance of words in the reviews. SVM is used to classify the sentiments.	SVM	IMDb	85.2%
10.	D Maity, S Kanakaraddi and S Giraddi, 2023 [41]	CNN is not utilized by the model only to extract local characteristics between texts but also employs bi-directional LSTM to collect semantic information globally for sentence context.	CNN, LSTM	IMDb	86.13%
11.	Proposed work	SAE is used for features extraction and LSTM is used as a classifier	SAE, LSTM	IMDb	87%

III. MOTIVATION

Social media has gained popularity in recent years as a tool for knowledge sharing around the world. Users exchange material via Facebook, Twitter, and other social media platforms, but they don't just share it; they also comment on it, expressing their thoughts, either positive or negative. This data cannot only be used in e-commerce but also what types of information is becoming popular in society. Sentiment analysis consider feelings and opinions rather than a count of mentions or comments.

Various standard Machine Learning and Deep Learning models have been utilized in sentiment analysis. The overall performance of standard models is not much accurate as mentioned in section II. Hence the need arises to design a hybrid of two models to improve accuracy. RNN is most common model for the analysis of sentiments. In standard RNN, vanishing gradient is the problem. Therefore, our proposed system used LSTM to overcome this problem. CNN needs multiple layers to acquire long-term dependencies while LSTM does not require that. Also, LSTM regulates the amount of newly contributed data to the cell.

IV. PROPOSED METHODOLOGY

An architecture is proposed for sentiment analysis that is a hybrid of LSTM and SAE. The proposed architecture concentrates on solving the issue of vanishing gradient that is frequently found in standard Recurrent Neural Network (RNN). The sentiments of the textual data are binary classified. The purpose of SAE is to provide dimensionality reduction. The deep learning model, LSTM, is used to classify the text sentiments and word level sentiment analysis is performed using google colab.

LSTM has been combined with auto-encoder to enhance the performance of standard LSTM for sentiment classification. Textual data is taken from the dataset and preprocessed. Preprocessing aids in the removal of noise or useless sections of data by converting all letters to lowercase, removing punctuation and stop words. After data cleaning, data is passed to the SAE for extracting useful features. Encoding and decoding are the two main phases of an auto-encoder. The encoding phase converts input features into a new representation, and the decoding phase precisely restores the original features to the new representation.

The stacked auto-encoder encodes the input data by encoder =Embedding(max_features,20)(inp) and encoder=LSTM (10,return_sequences=True)(encoder). The number of neurons in the encoding layer are 20. The data will be encoded in the embedding layer of stacked auto-encoder and provided to LSTM for classification of sentiments as the sandwich of stacked auto-encoder and LSTM is created. In this way, the sentiments would be classified.

Auto-encoder's main goal is to extract more usable and informative features from enormous data. Hence, feature dimensionality is reduced and there will be the less redundant data as well as it would be easy for DL model to analyze the sentiments.

LSTM model is then given the reduced features. LSTM layer of the model comprises of gates; input gate, forget gate, and output gate. The LSTM's gates control how a stream of data enters, is stored, and leaves the network. The extracted features from SAE are given to LSTM where short term memory and long term memory together will classify the sentiments of the text. The forget gate will remove the non-informative data and pass the data to candidate state, that contain the extracted features. The input will be processed from the input gate combined with the features present in the long term memory and hence on the basis of informative features, the output is predicted. If the value of output is close to 1, sentiment is positive and if the value is closer to 0, the sentiment is negative. In this way, the sentiment of the text are classified as either positive or negative.

The working flow of the whole architecture is illustrated in the Algorithm. When the text is classified, the performance of the whole model is checked out using evaluation metrics. Performance evaluation metrics that we used are confusion matrix, accuracy, precision, sensitivity, specificity, and F1 score. The accuracy of proposed hybrid approach is better than the accuracy of standard LSTM used for sentiment analysis. The proposed model's architecture is depicted in Fig 1.

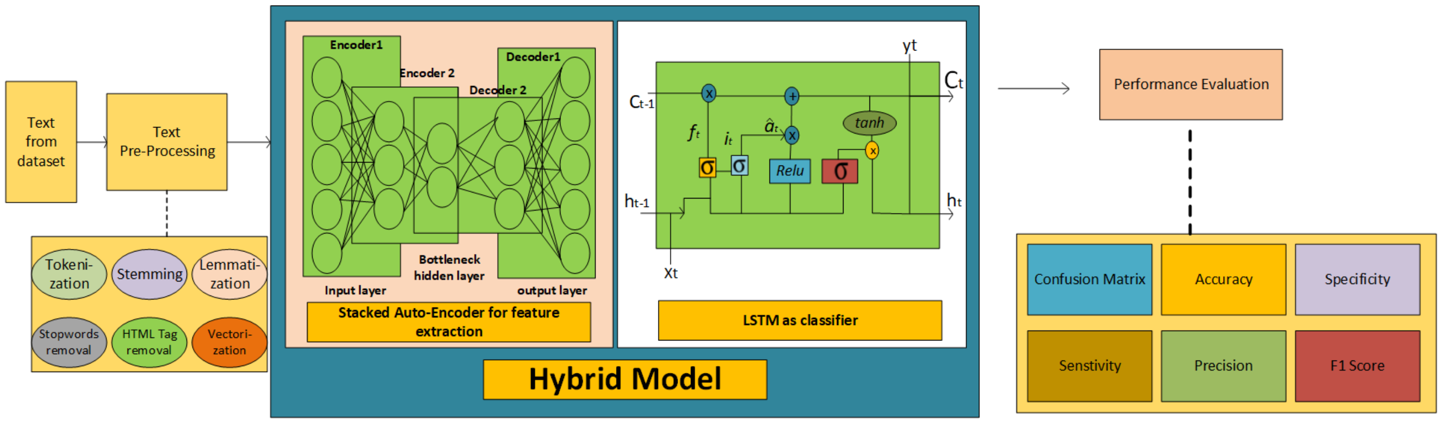


Fig. 1. Architecture of proposed hybrid model; text taken from dataset, preprocessed and passed on to the hybrid model where relevant informative features are extracted by Stacked Auto-encoder and passed to LSTM. LSTM will classify the sentiment of the text input on the basis of these extracted features. Further, the performance of classifier is evaluated using six classification evaluation metrics.

A. TEXT RE-PROCESSING

Text preprocessing is a method of cleaning and preparation of text data for use in models. Noise in form of emotions, punctuation, etc. is present in text data. Firstly, the text data is tokenized. Stop words, propositions, or the words that do not give us information about the sentiments, are removed. Tags for Parts of Speech (POS) are used and then these POS tags are used for entity recognition i.e. Named Entity Recognition (NER). NER allows us the quick recognition of significant aspects in document, i.e. people names, movie names, etc. POS tagging would be helpful for better vectorization of text and will make easy for SAE to extract the relevant features. Stemming and lemmatization are applied sequentially. Stemming will reduce the words to stem form. Text data is also lemmatized which will use the context of the words to reduce the word to canonical form. Although stemming reduces the word but it often changes the original word. Lemmatization will reduce the word and keep the original form of the word. After pre-processing, text data is vectorized that is transformed into vectors of fixed length as SAE and deep learning models can take only numbers as input.

B. TEXT FEATURE EXTRACTION USING STACKED AUTOENCODER – A NEURAL NETWORK

Auto encoder, a neural network that learns features. Encoding and decoding are the two stages of the process. In the encoding stage, the input data is transferred to a low-dimensional representation space to extract the most relevant feature which is then mapped back to the input space in the decoding phase. The hidden layer is referred to as a bottleneck since the autoencoder is commonly utilized for compression.The sigmoid and ReLu activation functions are used in the stacked auto-encoder. The reconstruction error is decreased between input-output data, the autoencoders learn significant features in the data. The number of output layer neurons of autoencoders is the same as the number of input layer neurons. Several layers of encoding and then an output layer of decoding are stacked to create SAE. A stacked autoencoder having two encoding and decoding layers are used. Autoencoders are most commonly trained as part of a bigger model in which input is replicated.

The architecture of the autoencoder model is constrained to a bottleneck at midpoint, in which the input is recreated. The model generates a fixed-length vector with a compressed version of the input data at the bottleneck. SAE's structure comprises numerous hidden layers, which allow it to represent complex high-dimensional functions and extract informative features. Information that is recorded within original space will be taken and transformed into a different space. For this particular task, the input representation contains some redundant information, which the transformation gets rid of. The original features are lost, but the new space has new features.The decoding layer will decode the new features into the original form and check either encoder has correctly extracted the informative features or not. Some of the features are selected while others are rejected by autoencoder. For instance, if the movie review is “The movie is awesome”, here the informative feature that is telling us about sentiment of the review is “awesome”. Thus, the SAE will select the feature “awesome” and reject the other non-informative words “The movie is”. Here, the goal is to determine the most accurate feature transformation, therefore, the stacked autoencoder is utilized so that the classifier LSTM will classify more accurately. The architecture of the SAE is shown in Fig 2. In the architecture, two encoders and decoders are used that will transform the actual features into reduced features effectively. SAE is very helpful in providing dimensionality reduction of data so that it will become easy for the classifiers to classify and will be easy for machine to perform any operations on text data.

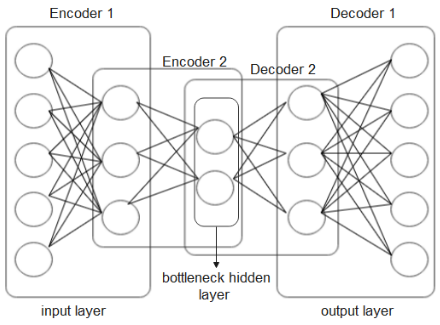


Fig. 2. SAE architecture

C. SENTIMENT CLASSIFICATION USING LSTM – A DEEP LEARNING MODEL

LSTM is a variant of standard RNN having the capability of learning long sequential data. LSTM has internal memory blocks, a gated mechanism to overcome the two common shortcomings of ordinary RNNs: vanishing gradient and exploding gradient. Memory cells having self-connection and specific multiplicative units are used in LSTM memory blocks to handle information flow. LSTM block comprises of three gates; input gate, output gate, and forget gate. Input is provided to the LSTM layer where the data to be removed from cell state is decided by the forget gate. The candidate state chooses the information to be written to the cell state, and the input gate chooses whether or not to do so. The data to be passed as an output hidden state is determined by output gate. The LSTM architecture is presented in Fig 3. In Fig 3, the input, output, and forget gates of the LSTM through time step t are (i_t), (o_t), and (f_t), respectively; (c_t) is the memory cell content; and (a_t) represents the candidate state determined in (4). The input to the input gate, output to the output gate, and data to forget through forget gate is calculated using (1), (2), and (3).

$$i_t = \text{Sigm}\,(W_{xi}\, x_t + U_{hi}\, h_{t-1} + b_i) \qquad (1)$$

$$o_t = \text{Sigm}\,(W_{xo}\, x_t + U_{ho}\, h_{t-1} + b_o) \qquad (2)$$

$$f_t = \text{Sigm}\,(W_{xf}\, x_t + U_{hf}\, h_{t-1} + b_f) \qquad (3)$$

In ordinary LSTM, tanh activation function is used in the candidate state. The vanishing gradient problem most frequently occurs with sigmoid and tanh functions. The initial layers’ weights and biases would not be adequately updated for each training session if the gradient is too tiny. Weights won’t be converging towards the global minima as a result. It can result in overall network inaccuracy because these early layers are frequently essential for identifying the fundamental components of input data. LSTM and GRU are more resistant to degradation of gradient descent than standard RNN [43]. When deep neural networks are being trained, utilizing the Rectified Linear Unit (ReLU) activation function can help us avoid the issue of gradient descent. ReLU changes input to its maximum value, which is either 0 or input. Input is transformed to 0 by ReLU if it is less than or equal to 0. Input gets transformed to the supplied input by ReLU if it is greater than 0. In the proposed system, tanh activation function in (4) is replaced with Relu and modified to (5).

$$\hat{a}_t = \tanh\,(W_{x\hat{a}}\; x_t + U_{h\hat{a}}\, h_{t-1} + b_{\hat{a}}) \qquad (4)$$

$$\hat{a}_t = \text{Relu}\,(W_{x\hat{a}}\; x_t + U_{h\hat{a}}\, h_{t-1} + b_{\hat{a}}) \qquad (5)$$

x_t, h_t, and h_{t-1} represent the hidden unit's input, final output, and preceding time step, respectively. The cell state vector is updated as shown in (6). W_{xo} and U_{ho} are weight matrices, b_o is bias term.

$$c_t = f_t * x_{t-1} + i_t * \hat{a}_t \qquad (6)$$

The output of the output gate o_t, (2) multiplied by the cell state c_t using tanh function in (7) for performing hidden state (h_t) of LSTM unit and sent to next sample in sequence.

$$h_t = o_t * \tanh\,(c_t) \qquad (7)$$

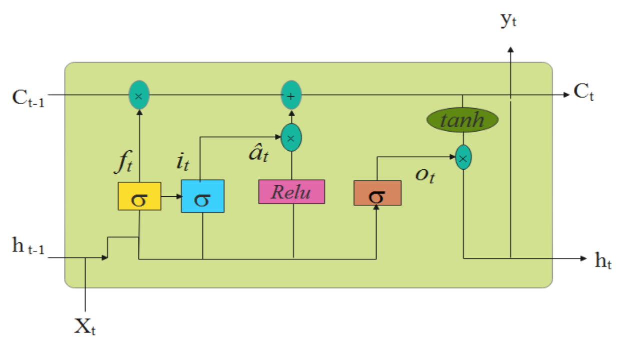


Fig. 3. LSTM architecture

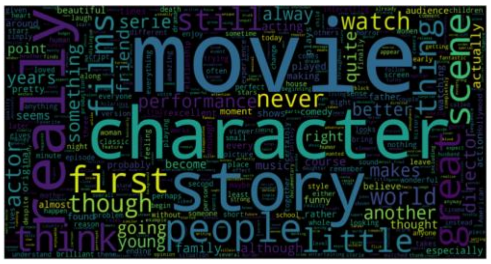
```
Algorithm:
Parameters initialization
    AE Parameters
    Inputs:
    -Input feature set [$x_1$,$x_2$,$x_3$,….,$x_n$]
    -Encoding activation function EAF
    -Decoding activation function DAF
    -Input weights Wi
    -Biases values bi
    LSTM Parameters
    -Input feature set [$\bar{x}_1$, $\bar{x}_2$, $\bar{x}_3$,…..,$\bar{x}_n$]
    -Weights for different gates are:
     Input gate: $W_{xi}$, $U_{hi}$
     Candidate state: $W_{x\hat{a}}$ , $U_{h\hat{a}}$
     Forget gate: $U_{hf}$, $W_{xf}$
     Output gate: $U_{ho}$, $W_{xo}$
    -Biases for different gates are:
     Input gate: $b_i$
     Candidate state: $b_{\hat{a}}$
     Forget state: $b_f$
    Output gate: $b_o$
Step 1: Create Auto-encoder model
Encoding
     -Encoded inputs $f(w)$ are computed by multiplying $x_n$ and $W_i$
     -Biased inputs $f(b)$ are computed by adding b$i$ to encoded inputs
     -Compute $f(p)$ applying $y = f(p) = k_e (W_x + b_h)$ using $f(w)$ and $f(b)$
Decoding
     -Compute decoded outputs $f(w^o)$ by multiplying $x_n$ and $W_i$
     -Compute biased outputs $f(b^o)$ by adding $b_t$ to decoded outputs
     -Calculate $g(q)$ applying $r = g(q) = k_d (W_y + b_x)$ using $f(w^o)$ and $f(b^o)$
Optimization
     -Optimize the value of cost function to reduce reconstruction error
      While (All layers trained)
     Output feature vector of AE is set as training vector of LSTM
Step 2: Train and validate model
    Inputs; $X_t$, $h_{t-1}$, and $c_{t-1}$ are passed to LSTM cell.
    -Compute input gate $i_t$ using eq. (1)
    -Compute output_gate_out $o_t$ using eq. (2)
    -Compute forget_gate_out $f_t$ using eq. (3)
    -Compute candidate state $\hat{a}_t$ using eq. (5)
```

```
        -input_gate_out = $i_t * \hat{a}_t$
        -Compute current cell state $c_t$ using
         $c_t = (c_{t-1} * \text{forget\_gate\_out}) + (\text{input\_gate\_out})$
        -Output of the LSTM cell $h_t$ is computed using
                $h_t = o_t * \tanh(c_t)$
        **While** stopping criteria did not met **do**
            **While** training for all instances **do**
            -Prepare a mini-batch features set as model input
            -Calculate loss function
            -Calculate the gradient using back propagation through time at time step t
            -Update weights and bias through back propagation algorithm
            **End while**
        **End while**
        **While (All layers trained)**
        **End while**
    **Step 3: Test model**
        -Test hyper-parameters with test dataset
         **return** Evaluate result in test dataset
```

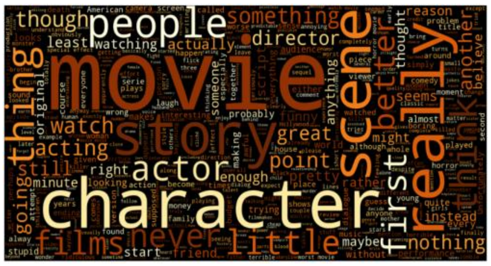
V. EXPERIMENTAL RESULTS

A. DATASET DESCRIPTION

The data collection is first step in every natural language processing and deep learning project. Real time datasets can be created by ourselves or we can take already created datasets available in the online UCI Repository or a huge platform for data scientists' viz. Kaggle. The dataset of movie reviews, IMDb, is used from the website of Kaggle. It consists of positive (50%) and negative (50%) reviews of various movies that have been classified as positive or negative. The positive and negative word cloud before preprocessing of data is shown in the Fig 5.



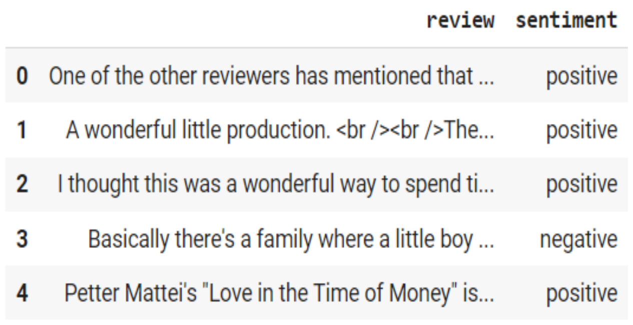
(a)



(b)

Fig. 5. Before pre-processing (a) positive word cloud (b) negative word cloud

The IMDb dataset consists of 50,000 instances and two columns; review and sentiment as shown in the Fig 6.

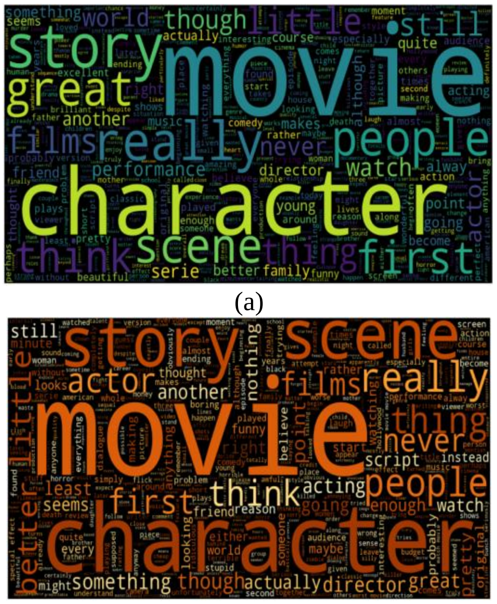


	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. <br /><br />The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive

Fig. 6: Looking over inside the dataset

B. IMPLEMENTATION DETAILS

To enhance the performance of simple DL models, the sentiment Analysis of IMDb dataset [59] provided by Kaggle is performed by using a hybrid model having autoencoder for feature extraction and LSTM as a classifier. Data is preprocessed and cleaned by removing punctuation, stop words, and all other irrelevant words which are not helpful in predicting the class of sentiment. Since machine learning and deep learning models take input in the form of number vectors but not in the form of text. The data that we utilized is text data, therefore, we need to convert text data to numbers which is the process of vectorization. Word embedding or vectorization is performed. When the data is cleaned, it is given to the model for prediction of sentiment. After preprocessing, the data will look like this as shown in Fig 7.



(a)

(b)

Fig. 7: After preprocessing (a) positive reviews (b) negative reviews

After preprocessing and vectorization, the data is passed to the stacked auto encoder for feature extraction and dimensionality reduction. Afterward, the output of the auto encoder is passed to the last layer of LSTM that will classify the sentiment of the reviews.

The dataset is one by one split up in different ratios of 60-40%, 80-20%, 70-30%, and 50-50% to evaluate the model's performance. The splitting of the dataset results in a testing dataset that is ideal for assessing the performance of a model fitted to the training dataset. When using different dataset segments for training, the model will not produce the same results. By rotating the training and validation sets, the effect of more and less training on the performance of model is checked out. For the training testing ratio of 90%-10%, the model has given better results. The encoder layer will compress the data and transform it into another compressed representation. The bottleneck layer will store the compressed representation that is later on utilized by the decoder. The decoder layer will then decode the data and check that either encoded data is correctly representing the actual data or not.

The input data having length of first 600 words of every sentence is provided to the two encoding layers. The encoding layer will extract the features automatically and pass them to LSTM merged in one of the decoding layer. On the basis of the extracted features, gates in the LSTM will make LSTM able to classify the sentiments of the text data. The globalmaxpooling 1D is used in the other decoding layer so that the data will obtain a form that is acceptable for dense layer merged with the decoder of the model. Hence, the output will be either positive or negative sentiment. The training of the model is performed in a way that firstly, hyper parameters tuning is performed. Various ranges of hyper parameters are checked out for which the model is performing with better accuracy and good fitting. The batches size for hyper parameter optimization is 128, 1000, and 1024. The range of number of neurons of LSTM layer is 10-64, epochs 5-70, the learning rate of 0.001, 0.01, and 0.002 respectively. The model is not showing better accuracy and good fitting for all the ranges of hyper parameter than the ones mentioned in the Table I. The Adam optimizer is used for optimization and as a loss function of binary cross-entropy is utilized. The hybrid model has chosen learning rate value by default. The model is showing better accuracy and good fitting for the hyper parameters illustrated in Table II. The batch size is 128, the number of epochs is 30, and validation ratio is 5% i.e. 0.05, and the major metric used is classification accuracy. For this whole scenario, the achieved accuracy of the model is 87%.

Table II. Setting of Hyper parameters for the model

Hyper parameters	**Value**
Optimizer	Adam
Batch size	128
Epochs	30
Number of layers	5
Number of Neurons	10
Loss Function	Binary cross entropy

C. COMPARATIVE ANALYSIS OF DIFFERENT TRAINING TESTING RATIOS

All the above mentioned (in section B) hyper parameters are utilized and different training and testing ratios are analyzed in terms of accuracy score and loss.

1) TRAINING TESTING RATIO OF 80/20

The number of epochs and various hidden layer neurons are used to train the model. The performance of the model is checked using 10, 15, and 30 neurons and 10, 15, and 30 epochs. The number of neurons and epochs are selected based on the accuracy score using trial and error mechanism. By increasing epochs than 30, the performance of the model particularly classification accuracy starts decreasing. Here, the results of 30 epochs with 10 neurons at an 80/20 ratio are written. Out of 20% testing data, 10% is for validation and remaining is for testing. When the model is trained for a training testing ratio of 80/20, the accuracy is 85% which can be seen in Fig 8.

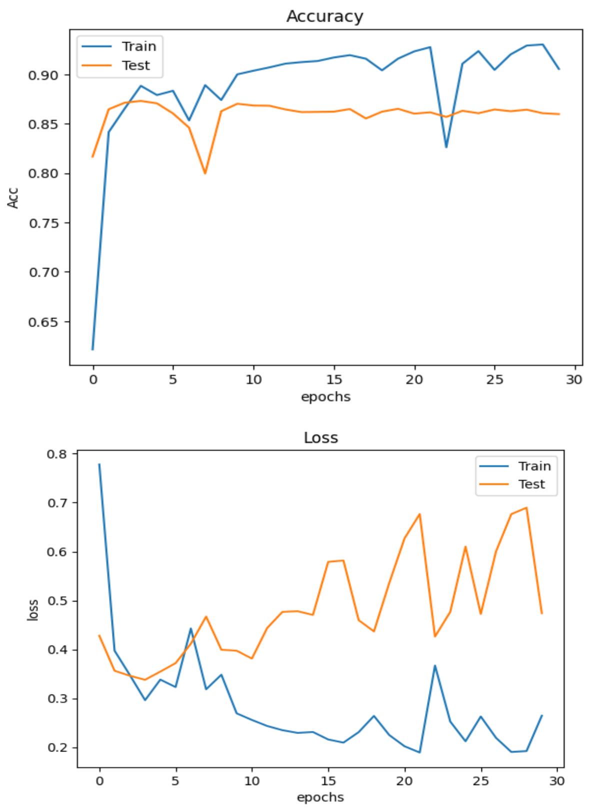


Fig. 8: Training and Testing accuracy and loss for ratio of 80/20

True Positive (TP) and True Negative (TN) values of the classification matrix illustrate how accurately the model has classified the sentiments. False Positive (FP) and False Negative (FN) values tell how many sentiments have not been predicted correctly by the sentiment classifier. Fig 9 depicts the confusion matrix.

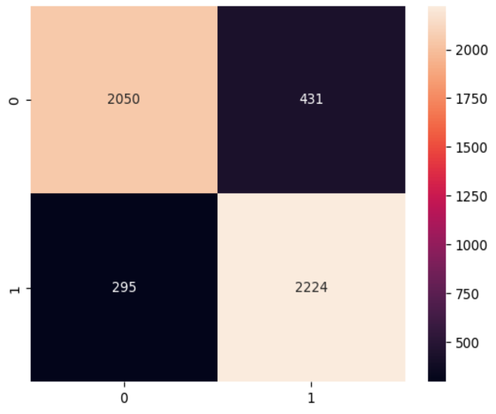
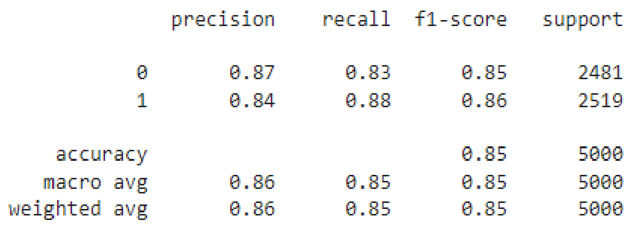


Fig. 9: Confusion Matrix

The recall score is greater than 0.5, shows that FN values are lower and the class is balanced. The values of evaluation metrics are depicted in Fig 10 of classification report. It tells us how better the model is performing.



```
              precision    recall  f1-score   support

           0       0.87      0.83      0.85      2481
           1       0.84      0.88      0.86      2519

    accuracy                           0.85      5000
   macro avg       0.86      0.85      0.85      5000
weighted avg       0.86      0.85      0.85      5000
```

Fig. 10: Classification Report

Receiver Operator Characteristic (ROC) curve, is an evaluation statistic used for binary classification tasks. By comparing True Positive Rate (TPR) to False Positive Rate (FPR) at different threshold levels, it is a probabilistic curve that effectively distinguishes signal from noise. It is indicated through AUC which is a ROC curve summary that how much effectively positive and negative classes can be discriminated by the classifier. The ROC score is 0.85 as shown in Fig. 11.

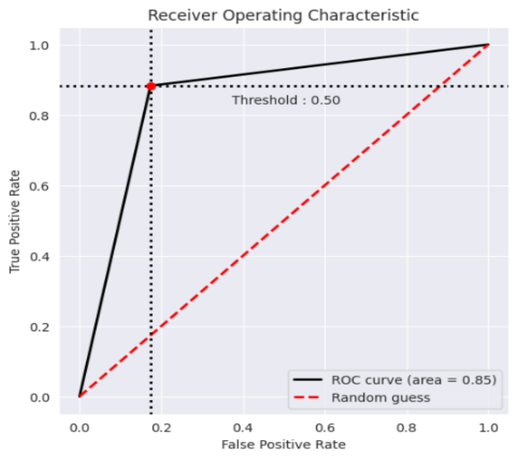
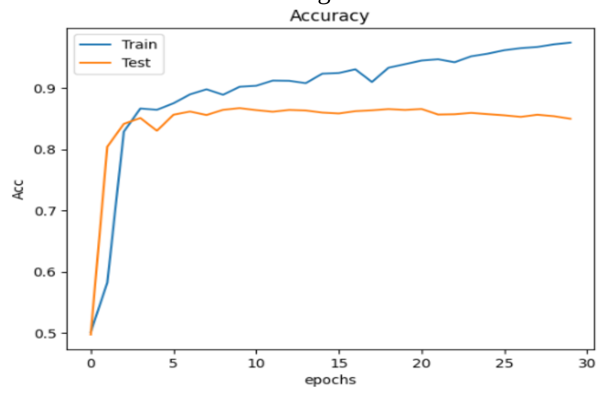


Fig. 11: ROC curve

2) TRAINING TESTING RATIO OF 70/30

The adjusted hyper parameters for the training of model at 70/30 ratio are 30 epochs and 10 neurons of LSTM layer. Out of 30% testing data, 15% is for validation and remaining is for testing. For the training testing ratio of 70/30, the accuracy of the model is 85% as shown in Fig 12.



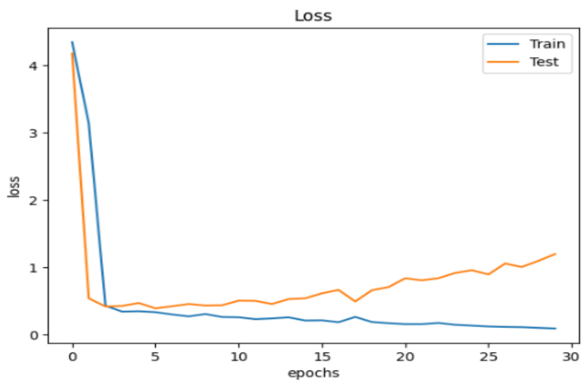


Fig. 12: Training and Testing accuracy and loss for ratio of 70/30

Fig 13 depicts the confusion matrix. If the precision value is 1, the classification will be ideally correct i.e. all the positive sentiments are classified as positive and negative ones are classified as negative. If the precision value is closer to 0, it would be less correct classification and if closer to 1, it would be more correct classification. TP and TN values i.e. 3247 and 3127 in Fig 13 are the correctly classified instances. The FP and FN values i.e. 461 and 665 are the ones classified incorrectly. The values of evaluation metrics are shown in Fig 14.

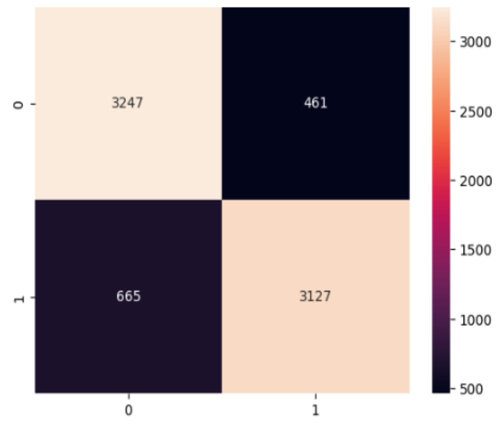
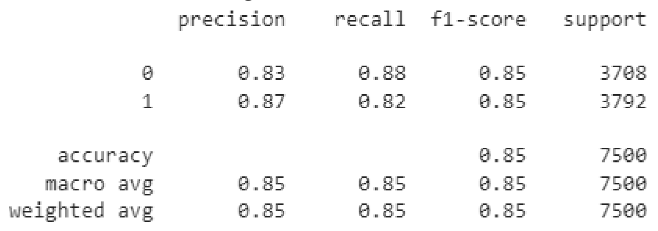


Fig 13: Confusion Matrix



```
              precision    recall  f1-score   support

           0       0.83      0.88      0.85      3708
           1       0.87      0.82      0.85      3792

    accuracy                           0.85      7500
   macro avg       0.85      0.85      0.85      7500
weighted avg       0.85      0.85      0.85      7500
```

Fig 14: Classification Report

In ROC curve, the True Positive Rate (TPR) tells that what proportion of positive class got correctly classified and False Positive Rate (FPR) tells about the proportion of negative class got incorrectly classified. The ROC score is 0.85 shown in Fig 15.

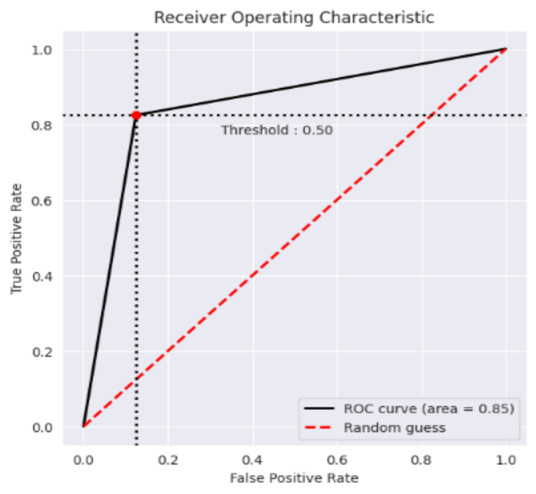


Fig. 15: ROC curve

3) TRAINING TESTING RATIO OF 60/40

The proposed model has been evaluated for numerous training testing ratios. Out of 40% testing data, 20% is for validation and remaining is for testing. For the training testing ratio of 60/40, the accuracy of the model is 85% as expressed in Fig 16.

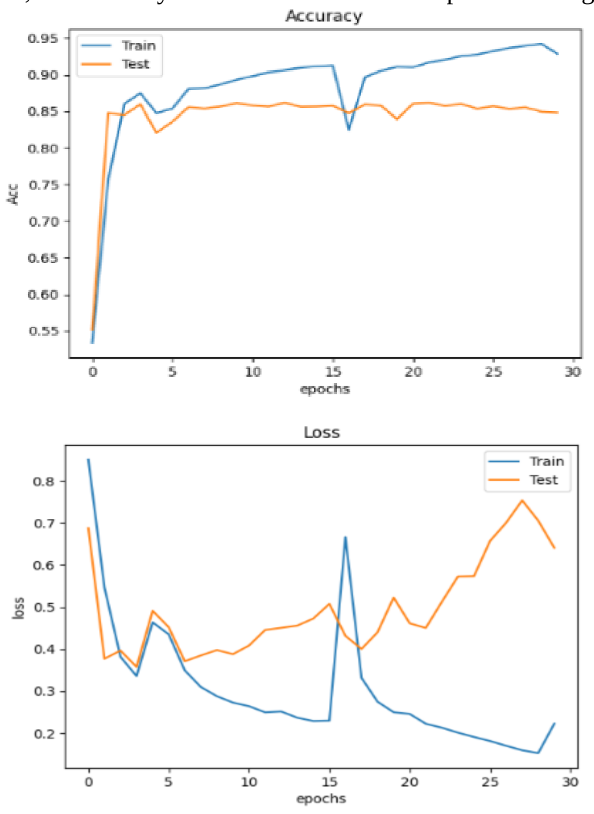


Fig. 16: Training and Testing accuracy and loss for ratio of 60/40

Fig 17 depicts the confusion matrix. The TP and TN values i.e. 4282 and 4229 in Fig 17 are the correctly classified instances. The FP and FN values i.e. 679 and 810 are the ones classified incorrectly. Higher the values of TP and TN, better is the classification performance of the model.

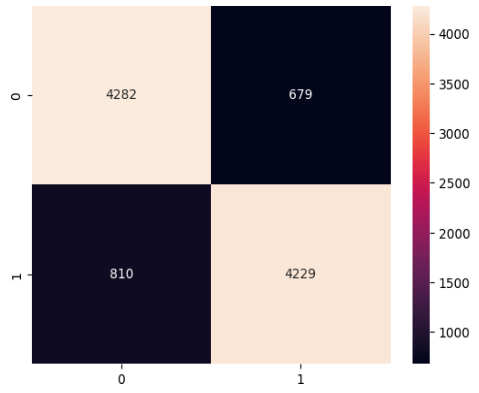
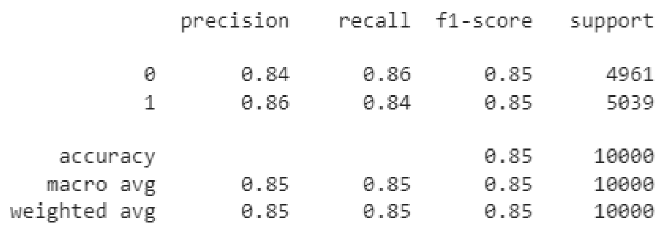


Fig. 17: Confusion Matrix

The support values are the number of actual occurrences of class in the dataset. Precision and F1 score are nearly equal to 1. Thus, the proposed model has correctly classified the sentiments of the data. The values of evaluation metrics are shown in the classification report as shown in Fig 18.



```
              precision    recall  f1-score   support

           0       0.84      0.86      0.85      4961
           1       0.86      0.84      0.85      5039

    accuracy                           0.85     10000
   macro avg       0.85      0.85      0.85     10000
weighted avg       0.85      0.85      0.85     10000
```

Fig. 18: Classification Report

In the proposed model, the Area Under Curve (AUC) score is 1, showing that the model has predicted positive and negative sentiments correctly. The ROC score is 0.85 as shown in Fig 19.

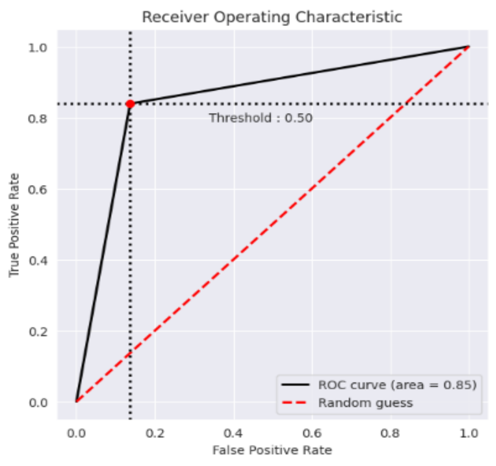


Fig. 19: ROC curve

4) TRAINING TESTING RATIO OF 50/50

The proposed model performance is checked at various training testing ratios. Out of 50% testing data, 25% is for validation and remaining is for testing. For the training testing ratio of 50/50, the accuracy is 80% as shown in Fig 20.

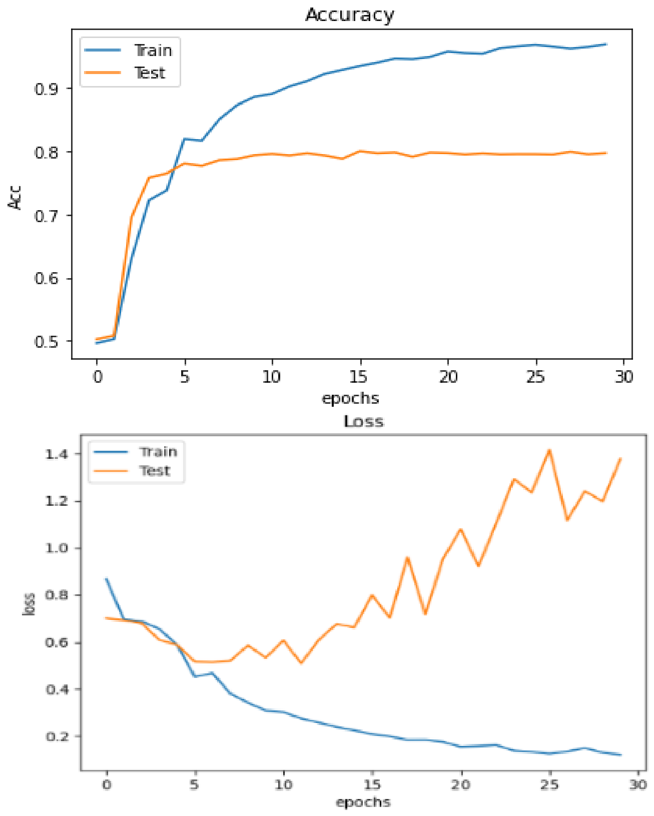


Fig. 20: Training and Testing accuracy and loss for ratio of 50/50

Confusion Matrix tells us about the performance of the model. TP and TN values i.e. 10044 and 9918 in Fig 21 are the correctly classified instances. The FP and FN values i.e. 2439 and 2599 are the ones classified incorrectly. Fig 21 depicts the confusion matrix.

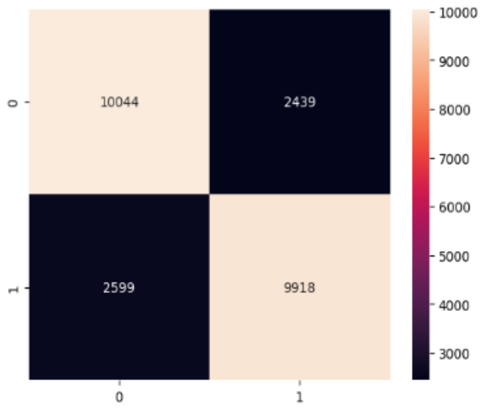
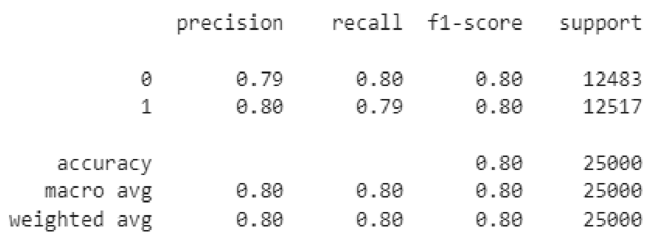


Fig. 21: Confusion Matrix

The values of evaluation metrics are shown in the classification report as shown in Fig 22.



```
              precision    recall  f1-score   support

           0       0.79      0.80      0.80     12483
           1       0.80      0.79      0.80     12517

    accuracy                           0.80     25000
   macro avg       0.80      0.80      0.80     25000
weighted avg       0.80      0.80      0.80     25000
```

Fig. 22: Classification Report

AUC measures the ability of the classifier while ROC distinguishes between positive and negative classes. If AUC=0, model is predicting all negative as positive and all positive ones as negative. If AUC>0.5 and AUC<1, the model would be able to distinguish between positive and negative sentiments but TP and TN values will be more than FP and FN. ROC score is 0.87 and AUC is 1 as shown in Fig 23.

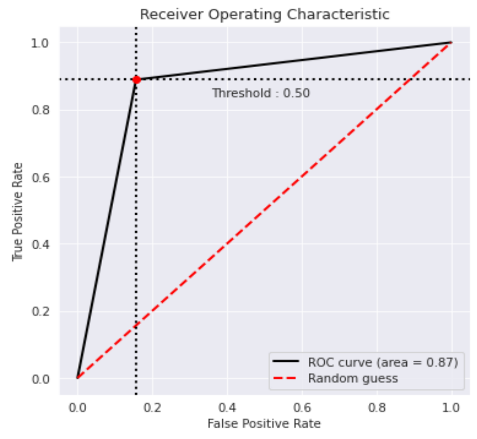
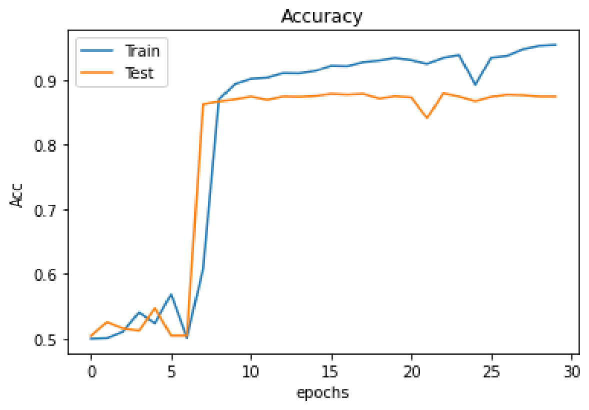


Fig. 23: ROC curve

5) TRAINING TESTING RATIO OF 90/10

Adam optimizer for optimization and binary cross entropy is used as loss function. The batch size 128, the number of epochs 10, the validation ratio 10%. Out of 10% testing data, 5% is for validation and remaining is for testing. The accuracy of the model is 87%. Training loss is decreasing while training accuracy is increasing as illustrated in Fig 24.



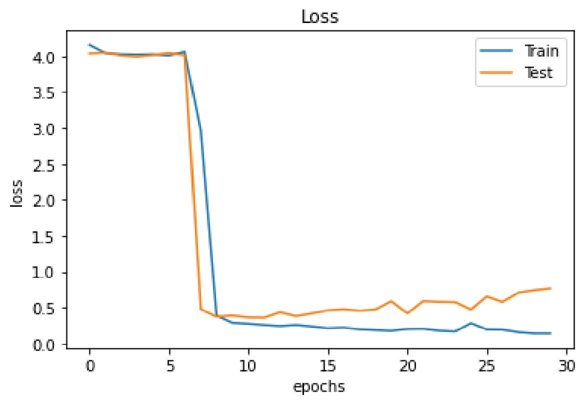


Fig. 24: Training and Testing Accuracy and Loss

TP and TN values i.e. 1045 and 1121 in Fig 29 are the correctly classified instances. The FP and FN values i.e. 194 and 140 are the ones classified incorrectly. Fig 25 depicts the confusion matrix.

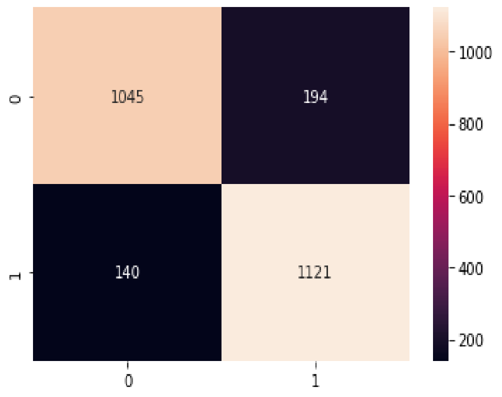
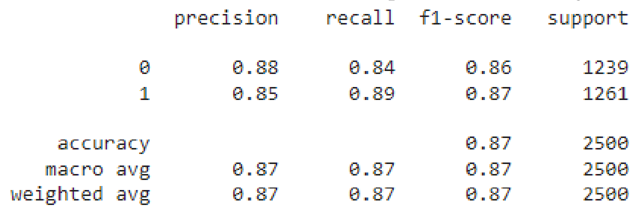


Fig 25: Confusion Matrix

The precision score is 0.88 and 0.85 respectively as shown in Fig. 26, showing that most of the positive and negative sentiments are predicted accurately. Recall score > 0.5, showing that classifier has less number of FN values, class is balanced and hyper parameters are tuned accurately. F1 score is 0.86 and 0.87 respectively that is nearly equals to 1. Therefore, most of the sentiments are predicted correctly.



```
              precision    recall  f1-score   support

           0       0.88      0.84      0.86      1239
           1       0.85      0.89      0.87      1261

    accuracy                           0.87      2500
   macro avg       0.87      0.87      0.87      2500
weighted avg       0.87      0.87      0.87      2500
```

Fig 26: Classification Report

The ROC score is 0.87 as shown in Fig 27 which shows that the proposed hybrid model performed well and classified positive and negative classes in a better way. In proposed model, AUC=1, so the model has correctly predicted the positive and negative sentiments.

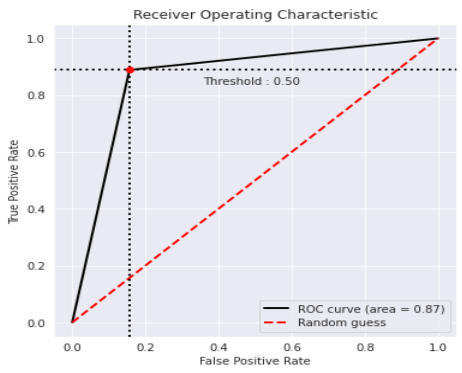


Fig. 27: ROC curve

VI. COMPARISON OF PERFORMANCE OF PROPOSED MODEL WITH RESPECT TO DIFFERENT RATIOS AND K-FOLD CROSS VALIDATION

The hybrid model has performed best at 90/10 ratio of training and testing respectively having classification accuracy of 87%. At other training testing ratios of 80/20, 70/30, 60/40, and 50/50, model has less accuracy. The whole comparison of results on the basis of all classification metrics in aspect to training testing ratios is shown in Fig 28.

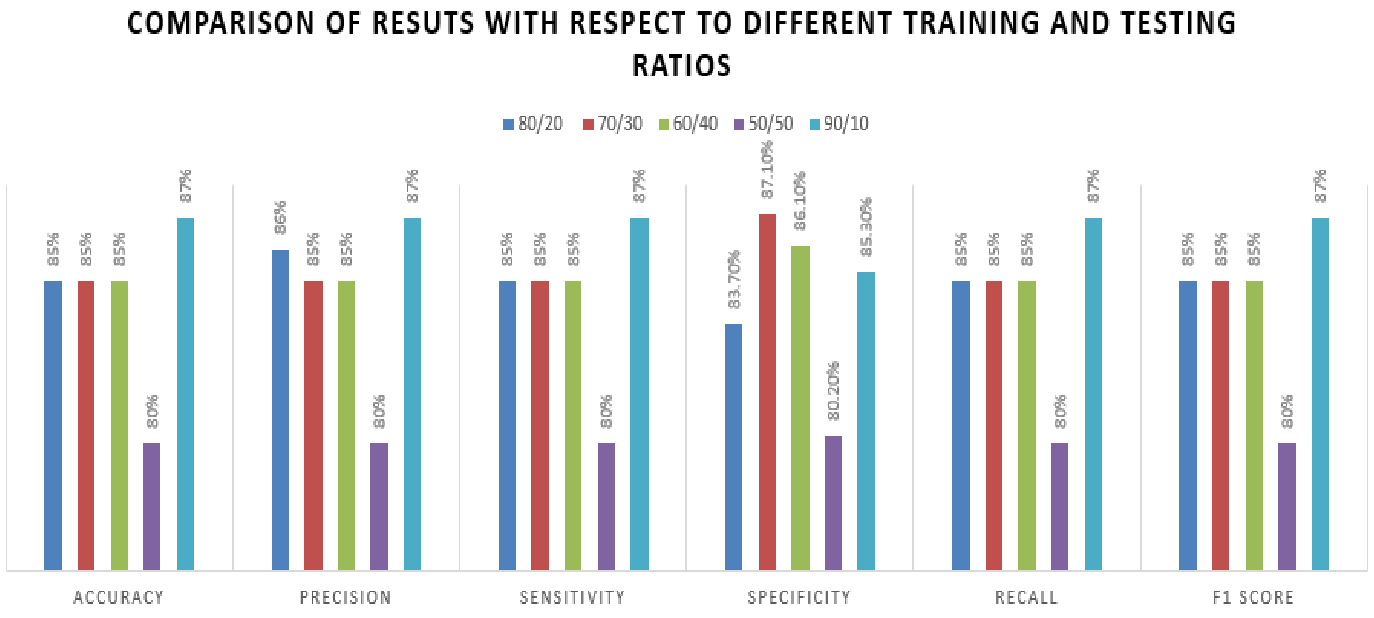


Fig. 28: Comparison of results with respect to different training and testing ratios

K-fold cross validation is the most popular techniques frequently employed by data scientists.. It is a method of data partitioning that enables you to make the most of your dataset when creating a more comprehensive model. The entire dataset is randomly divided into independent k-folds without replacing the instances. One fold is utilized for performance assessment, and k-1 folds are utilized for model training. We repeat the process k times(iterations) to get the estimates of k performance for every iteration. Then, we obtain the mean of k performance estimates. The iterations k do vary; 20% of the test set is withheld when k=5, 10% of the test set is always withheld when k = 10. Here, 10 fold cross validation has been utilized so 10% of the test data is withheld. The 10 fold cross validation is performed using proposed model having stacked auto-encoder and LSTM. Results are given in the Table III and compared with the other approaches also.

Table III. Comparison of proposed model's performance with 10 fold cross validation and past approaches

Serial No.	Model	Classification Accuracy	Precision	Recall	F1 score	Specificity
1.	Proposed Model	87%	87%	87%	87%	85.30%
2.	Proposed Model with k-fold cross validation (k=10)	86.7%	86%	86.7%	86%	85%
3.	Ahmed K. et al., 2022 [35]	SAE 85.1% SVM 82.9%	SAE 87.9% SVM 82.7%	SAE 86.9% SVM 82.9%	SAE 87.4% SVM 82.7%	-
4.	CH Kumar and RS Kumar, 2022 [37]	83.5%	-	-	-	-
5.	SichangSu, 2022 [40]	85.2%	85.2%	100%	92%	-
6.	D Maity, S Kanakaraddi and S Giraddi, 2023 [41]	86.13%	-	-	-	-

Experimental results have been discussed here. In comparison to other basic models, the hybrid model's classification accuracy is evaluated for various training testing ratio of dataset. It is found that the proposed hybrid model’s performance is better as compared to the simple deep learning and machine learning models.

VII. CONCLUSION

Numerous machine learning and deep learning models for sentiment analysis are proposed and tested lately. In this work, the hybrid model is trained for sentiment classification of movie reviews. The information gathered for training is 90% and for testing 10% data has been taken. The accuracy of the proposed model is 87%. Hence, the suggested hybrid model is better for analysis of sentiment as evidenced by the proposed model's increased accuracy in comparison with simple deep learning models. The model has more accurately classified both positive and negative opinions expressed in movie reviews. The shortcoming of the model is that it has not been elaborated for multiclass classification problem ignoring neutrality or ambivalence.

In future, sentiment analysis can be performed using hybrid of other machine learning and deep learning models. Meanwhile, tertiary classification of sentiments can also be performed i.e. positive, negative and neutral. The proposed model can be used in various other fields; Human Computer Interaction (HCI), statistical analysis, digital marketing.

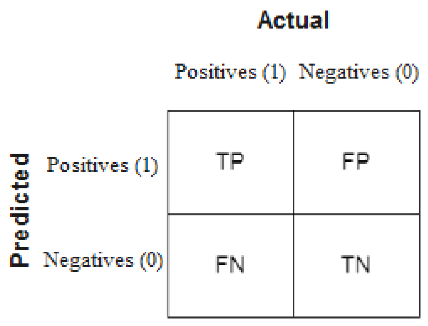
APPENDIX

PERFORMANCE EVALUATION METRICS

The evaluation of classifier performance is carried out using confusion matrix, accuracy, precision, sensitivity, specificity and F1 score.

1) CONFUSION MATRIX

An evaluation metric for classification when the result can be two or more classes is confusion matrix.



2) CLASSIFICATION ACCURACY

Accuracy is defined as the proportion of the model’s total predictions that were correctly predicted.

3) PRECISION

The percentage of correctly foreseen positive predictions that occur is known as precision.

4) SENSITIVITY

The fraction of actual positive cases that were positively predicted is known as sensitivity (or true positive). Sensitivity is also termed as “recall”.

5) SPECIFICITY

The fraction of actual negative cases that were negatively predicted is known as specificity (or true negative).

6) F1 SCORE

The harmonic mean of recall and precision is F1 score. It represents precision and recall.

REFERENCES

[1] A. Hussain and E. Cambria, "Semi-supervised learning for big social data analysis," *Neurocomputing,* vol. 275, p. 1662–1673, 2018.

[2] Y. Xia, E. Cambria, A. Hussain and H. Zhao, "Word polarity disambiguation using bayesian model and opinion-level features," *Cognitive Computation,* vol. 7, p. 369–380, 2015.

[3] I. Chaturvedi, E. Ragusa, P. Gastaldo, R. Zunino and E. Cambria, "Bayesian network based extreme learning machine for subjectivity detection," *Journal of The Franklin Institute,* vol. 355, p. 1780–1797, 2018.

[4] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," *Advances in neural information processing systems,* vol. 27, p. 2177–2185, 2014.

[5] S. Lai, K. Liu, S. He and J. Zhao, "How to generate a good word embedding," *IEEE Intelligent Systems,* vol. 31, p. 5–14, 2016.

[6] M. Li, Q. Lu, Y. Long and L. Gui, "Inferring affective meanings of words from word embedding," *IEEE Transactions on Affective Computing,* vol. 8, p. 443–456, 2017.

[7] M. Song, H. Park and K.-s. Shin, "Attention-based long short-term memory network using sentiment lexicon embedding for aspect-level sentiment analysis in Korean," *Information Processing & Management,* vol. 56, p. 637–653, 2019.

[8] J. Pennington, R. Socher and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP),* 2014.

[9] A. Voulodimos, N. Doulamis, A. Doulamis and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Computational intelligence and neuroscience,* vol. 2018, 2018.

[10] I. Chaturvedi, R. Satapathy, S. Cavallari and E. Cambria, "Fuzzy commonsense reasoning for multimodal sentiment analysis," *Pattern Recognition Letters,* vol. 125, p. 264–270, 2019.

[11] A. Khatua, A. Khatua and E. Cambria, "A tale of two epidemics: Contextual Word2Vec for classifying twitter streams during outbreaks," *Information Processing & Management,* vol. 56, p. 247–257, 2019.

[12] F. Z. Xing, E. Cambria and R. E. Welsch, "Intelligent asset allocation via market sentiment views," *ieee ComputatioNal iNtelligeNCe magaziNe,* vol. 13, p. 25–34, 2018.

[13] L. Zhang, S. Wang and B. Liu, "Deep learning for sentiment analysis: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery,* vol. 8, p. e1253, 2018.

[14] Y. Mehta, N. Majumder, A. Gelbukh and E. Cambria, "Recent trends in deep learning based personality detection," *Artificial Intelligence Review,* p. 1–27, 2019.

[15] A. Chatterjee, U. Gupta, M. K. Chinnakotla, R. Srikanth, M. Galley and P. Agrawal, "Understanding emotions in text using deep learning and big data," *Computers in Human Behavior,* vol. 93, p. 309–317, 2019.

[16] G. Liu and J. Guo, "Bidirectional LSTM with attention mechanism and convolutional layer for

text classification," *Neurocomputing,* vol. 337, p. 325–338, 2019.

[17] S. M. Rezaeinia, R. Rahmani, A. Ghodsi and H. Veisi, "Sentiment analysis based on improved pre-trained word embeddings," *Expert Systems with Applications,* vol. 117, p. 139–147, 2019.

[18] A. M. Ramadhani and H. S. Goo, "Twitter sentiment analysis using deep learning methods," in *2017 7th International annual engineering seminar (InAES)*, 2017.

[19] M. S. Akhtar, A. Ekbal and E. Cambria, "How intense are you? Predicting intensities of emotions and sentiments using stacked ensemble [application notes]," *IEEE Computational Intelligence Magazine,* vol. 15, p. 64–75, 2020.

[20] M. Sharma, I. Kandasamy and W. B. Vasantha, "Comparison of neutrosophic approach to various deep learning models for sentiment analysis," *Knowledge-Based Systems,* vol. 223, p. 107058, 2021.

[21] M. Islam, A. Anjum, T. Ahsan and L. Wang, "Dimensionality reduction for sentiment classification using machine learning classifiers," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019.

[22] A. Singh, B. S. Prakash and K. Chandrasekaran, "A comparison of linear discriminant analysis and ridge classifier on Twitter data," in *2016 International Conference on Computing, Communication and Automation (ICCCA)*, 2016.

[23] Q. Pu and G.-W. Yang, "Short-text classification based on ICA and LSA," in *International Symposium on Neural Networks*, 2006.

[24] A. Abbasi, S. France, Z. Zhang and H. Chen, "Selecting attributes for sentiment classification using feature relation networks," *IEEE Transactions on Knowledge and Data Engineering,* vol. 23, p. 447–462, 2010.

[25] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013.

[26] C. Baziotis, N. Pelekis and C. Doulkeridis, "Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis," in *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*, 2017.

[27] S. Xiong, H. Lv, W. Zhao and D. Ji, "Towards Twitter sentiment classification by multi-level sentiment-enriched word embeddings," *Neurocomputing,* vol. 275, p. 2459–2466, 2018.

[28] K. Baktha and B. K. Tripathy, "Investigation of recurrent neural networks in the field of sentiment analysis," in *2017 International Conference on Communication and Signal Processing (ICCSP)*, 2017.

[29] X. Ouyang, P. Zhou, C. H. Li and L. Liu, "Sentiment analysis using convolutional neural network," in *2015 IEEE international conference on computer and information technology; ubiquitous computing and communications; dependable, autonomic and secure computing; pervasive intelligence and computing*, 2015.

[30] R. Sharma, A. Somani, L. Kumar and P. Bhattacharyya, "Sentiment intensity ranking among adjectives using sentiment bearing word embeddings," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017.

[31] G. Rao, W. Huang, Z. Feng and Q. Cong, "LSTM with sentence representations for document-level sentiment classification," *Neurocomputing,* vol. 308, p. 49–57, 2018.

[32] X. Zhou, X. Wan and J. Xiao, "Attention-based LSTM network for cross-lingual sentiment classification," in *Proceedings of the 2016 conference on empirical methods in natural language processing*, 2016.

[33] P. Astya and others, "Sentiment analysis: approaches and open issues," in *2017 International Conference on computing, Communication and automation (ICCCA)*, 2017.

[34] N. Jnoub, F. Al Machot and W. Klas, "A domain-independent classification model for sentiment analysis using neural models," *Applied Sciences,* vol. 10, p. 6221, 2020.

[35] K. Ahmed, M. I. Nadeem, D. Li, Z. Zheng, Y. Y. Ghadi, M. Assam and H. G. Mohamed, "Exploiting Stacked Autoencoders for Improved Sentiment Analysis," *Applied Sciences,* vol. 12, p. 12380, 2022.

[36] A. Fan, T. Lavril, E. Grave, A. Joulin and S. Sukhbaatar, "Addressing some limitations of transformers with feedback memory," *arXiv preprint arXiv:2002.09402,* 2020.

[37] C. H. Kumar and R. S. Kumar, "Natural Language Processing of Movie Reviews to Detect the Sentiments using Novel Bidirectional Encoder Representation-BERT for Transformers over Support Vector Machine," *Journal of Pharmaceutical Negative Results,* p. 619–628, 2022.

[38] C. Sun, X. Qiu, Y. Xu and X. Huang, "How to fine-tune bert for text classification?," in *Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18*, 2019.

[39] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *Advances in neural information processing systems,* vol. 32, 2019.

[40] S. Su, "Sentimental Analysis Applied on Movie Reviews," *Journal of Education, Humanities and Social Sciences,* vol. 3, p. 188–195, 2022.

[41] D. Maity, S. Kanakaraddi and S. Giraddi, "Text Sentiment Analysis based on Multichannel Convolutional Neural Networks and Syntactic Structure," *Procedia Computer Science,* vol. 218, p. 220–226, 2023.

[42] K. Dhola and M. Saradva, "A Comparative Evaluation of Traditional Machine Learning and Deep Learning Classification Techniques for Sentiment Analysis," in *2021 11th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, 2021.

[43] S.-H. Noh, "Analysis of gradient vanishing of RNNs and performance comparison," *Information,* vol. 12, p. 442, 2021.

IQRA KANWAL received her MCS degree in 2019 and MS Computer Science degree from University of Haripur, KPK, Pakistan in 2022, respectively. Her current research areas are sentiment analysis, machine learning, deep learning, and internet of things. Her major research interests include artificial intelligence and related fields.

FAZLI WAHID is currently working as Assistant Professor in Department of Information Technology, The University of Haripur, Pakistan. He has previously worked as Assistant Professor of computer science in University of Lahore, Pakistan. He completed his PhD in computer science from Universiti Tun Hussein Onn Malaysia in 2020. He received MS Computer Science degree from SZABIST, Islamabad, Pakistan in 2015 and BS Computer Science degree from University of Malakand, Pakistan in 2006. His research areas include machine learning, deep learning, medical imaging, artificial intelligence and related fields. Related to all these areas, he has published more than 40 research articles in well reputed journals and conferences. His areas of interest are energy consumption prediction, optimization, and management using multilayer perceptron, Artificial Bee Colony, Ant Colony, Swarm Intelligence, and other Machine Learning Techniques.

SIKANDAR ALI received the Ph.D. and post doctorate from the China University of Petroleum, Beijing, in 2019 and 2021 respectively. He is currently working as an Assistant Professor. He has authored over seventy articles in highly-cited journals and conferences. His research interests include Business process management, Anomaly detection in multi sensor system, software outsourcing partnership, software testing and test automation, bug prediction, bug fixing, software incidence classification, source code transformation, agile software development, and global software engineering. He served as a technical program committee member for more than twenty conferences and act as a reviewer for many well reputed Journals. He also organizes many Special Issues, Currently, he is working as an academic editor for Mobile Information Systems, Journal of Computer Information Systems, and Scientific Programming Journals.

ATEEQ UR REHMAN is currently working as an Associate professor in the Department of Information Technology, The University of Haripur, Pakistan. He received his PhD degree from the University of Southampton, the UK in 2017. As a PhD student, he worked in Southampton wireless research group of the University of Southampton, where he focused on reliable data transmission in Cognitive Radio Networks. He is a recipient of several academic awards, such as the Pakistan government Faculty development program, Islamic University of Technology (OIC) Dhaka, Bangladesh Distinction award and Higher education Commission Pakistan OIC scholarship for undergrad studies. His main research interests are next-generation wireless communications and cognitive radio networks, IoT, IoVT and blockchain technology and privacy-preserved Machine Learning, particularly in healthcare, and smart cities.

AHMED ALKHAYYAT received the B.Sc. degree in electrical engineering from AL KUFA University, Najaf, Iraq, in 2007, the M.Sc. degree from the Dehradun Institute of Technology, Dehradun, India, in 2010, and PhD from Cankaya University, Ankara, Turkey, in 2015. He contributed in organizing a several IEEE conferences, workshop, and special sessions. He is currently a dean of international relationship and manager of the word ranking in the Islamic university, Najaf, Iraq. To serve my community, I acted as a reviewer for several journals and conferences. His research interests include IoT in the health-care system, SDN, network coding, cognitive radio, efficient-energy routing algorithms and efficient-energy MAC protocol in cooperative wireless networks and wireless body area network, as well as cross-layer designing for self-organized network.