IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 13 March 2024, accepted 6 May 2024, date of publication 14 May 2024, date of current version 24 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3400992

RESEARCH ARTICLE

Children's Sentiment Analysis From Texts by Using Weight Updated Tuned With Random Forest Classification

AZHAR AHMED BILAL[1,2], O. AYHAN ERDEM[3], AND SINAN TOKLU[3]

[1]Department of Computer Engineering, Graduate School of Natural and Applied Sciences, Gazi University, 06560 Ankara, Turkey
[2]College of Computer Science and Information Technology, Kirkuk University, Kirkuk 36001, Iraq
[3]Department of Computer Engineering, Faculty of Technology, Gazi University, 06560 Ankara, Turkey

Corresponding author: Azhar Ahmed Bilal (aahmed.bilal@gazi.edu.tr)

ABSTRACT Sentimental Analysis is considered a computational strategy that helps in identifying and assessing the emotions of people via text documents. Tools and different methods have been adopted for determining both positive and negative emotions in the form of text data analytics by using Machine and Deep Learning techniques. Experimentally, it has been shown that the accuracy of existing text classification models such as Bi-LSTM, Decision Tree, and Ensemble Classifiers is limited by poor quality data, inappropriate hyperparameter tuning, and model-specific bias levels. Additionally, these models are prone to overfitting, high computational overhead, and longer training time. To overcome these limitations, we proposed a hybrid binary classification framework by combining Deep sequential features with the Random Forest (RF) technique. The approach is implemented in four phases: Initially, data preprocessing is performed by employing a Vader sentiment package. In the second step, the deep Long Short Term Memory (LSTM) model was employed to extract deep sequential features corresponding to sad and happy emotions. In the third phase, a bi-orthogonalization algorithm with principal component Analysis (PCA) and Singular Value Decomposition (SVD) was employed to minimize the redundancy and maximize the relevance of extracted features. Finally, a five-fold cross-validation technique was implemented to discriminate sad and happy emotions using the Random Forest (RF) algorithm. Eventually, a grid search approach was implemented for hyperparameter tuning and results were compared with five baseline algorithms (Vanilla LSTM (VLSTM), Support Vector Machine (SVM), Gradient Boosting Machine (GBM), Naïve Bayes (NB), Ada Boost Algorithm (ABA). The experimental outcomes revealed that the proposed model achieved an accuracy rate of 99.631% on the 4000 stories dataset which was superior to all five state-of-the-art methods with a margin of **4.63%, 10.7%, 19.44%, 21%, and 56.5%,** respectively. Interestingly, the proposed model realized improved results in terms of other conventional performance metrics also such as precision, recall, specificity, and time complexity. Overall, the proposed model has great potential in educational institutions, child psychology research, and child-friendly content moderation, generally helping in the understanding of the emotions and experiences of children in the digital realm.

INDEX TERMS Long short term memory (deep LSTM), natural language processing (NLP), principal component analysis (PCA), sentiment analysis (SA), singular value decomposition (SVD).

I. INTRODUCTION

The habit of reading books changes the way of thinking ability of a person and makes numerous changes within humans. This is because the person is completely involved with the characters and messages given in stories [1], especially when children begin reading books, their emotions will change based on the type of book they prefer. There are different emotions such as (happiness, sorrow, neglect loneliness, and trauma) that humans feel while reading,

The associate editor coordinating the review of this manuscript and approving it for publication was Agostino Forestiero.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

and this leads to creating a relationship between the stories that are mentioned in books and humans [2]. Since books are considered to be a medium for channeling emotions, internet technology has performed SA (Semantic Analysis) to interpret the type of emotions that the user gets after finishing reading [3]. SA performs the process of identification, extraction, and characterization of sentiments from the normal text with the help of NLP (Natural Language Processing), ML (Machine Learning), and DL (Deep Learning) methods [4]. Several approaches have been utilized in SA in terms of class-specific informative feature extraction and the application of appropriate classification techniques. In the feature extraction step, models determine the valuable characteristics of input text to improve model performance and interpretability. This step is crucial because it transforms raw data into a more concise and valuable format for analysis and modeling. By identifying and isolating key features from the original data, we can simplify its complexity, eliminate distractions, and pinpoint significant patterns or traits that are essential for sentiment data modeling. Existing text-based feature extraction methods such as Bag-of-Words (BoW) [5], Term Frequency-Inverse Document Frequency (TF-IDF) [6], Word Embeddings [7], and N-Gram Models [8] have been successfully implemented in various studies to identify the true semantic patterns. For example: a multiobjective, optimization-based weighted voting approach was used to assign accurate weight scores to the classifiers and predict the true labels [9]. The proposed ensemble method is static classifier selection with majority voting error, forward search and multi-objective differential evolution algorithm. By adopting the static classifier selection approach, our proposed ensemble method integrates Bayesian logistic regression, naïve Bayes, linear discriminant analysis, logistic regression, and support vector machines as the base learners whose performance in terms of precision and recall values determines the weight adjustment. The performance of the proposed model was limited by overfitting issues because of model complexity and the use of humongous parameters. Few other classification algorithms including naïve Bayes (NB), multinomial NB, Bernoulli NB, SVM, parametric SVM, logistic regression, decision tree, and stochastic gradient descent (SGD) were separately used in emotion classification. The primary limitation of these approaches was poor adaptability because of the application of handcrafted features.

Mowlaei et al. [10] suggested an approach for adaptive aspect-based lexicons for sentiment classification. The authors developed two strategies: (1) Statistics, and (2) Genetic Algorithm (GA) for designing dynamic lexicons and improving the classification accuracy [10]. In this work, the dynamic lexicon was automatically updated and provided more precise grading for context-related concepts [11]. Sentiment research has already been employed in a variety of domains, including hotels, airlines, healthcare, and the stock market [12], [13].

Sentiment analysis and natural language processing are related to a variety of concerns, including people's informal writing styles, sarcasm, irony, and language-specific challenges. Many words in different languages have meanings and orientations that vary depending on the context and domain in which they are used. As a result, there are few tools and resources available for each language. Furthermore, semantic analysis approaches sometimes require significant volumes of annotated data for training, making them resource-intensive and restricting their application to languages or domains with limited resources [14]. Another restriction is the difficulty in capturing long context-dependent semantics, particularly in sophisticated verbal phenomena like sarcasm, irony, and metaphor [15]. These models, especially when large and complex, are susceptible to overfitting, particularly when trained on small or noisy datasets [16]. Furthermore, semantic analysis approaches may struggle with out-of-vocabulary words or uncommon linguistic formulations that were not encountered during training, reducing their robustness and generalization capabilities.

Deep learning models, such as convolutional neural networks (CNNs), belief networks, and recurrent neural networks (RNNs), offer several key advantages over traditional machine learning methods [17]. Firstly, deep learning architectures are capable of automatic learning from the hierarchical representations of data, which allows them to model complex patterns and relationships within the text data. Furthermore, deep learning processes are very efficient at dimensional data management, e.g. text, by efficiently dealing with the huge complexity caused by semantic analysis tasks. Long Short-Term Memory (LSTM) is one of the popular variants of RNN that resolves the issue of vanishing gradients, a common problem in neural networks [18]. This enhancement allows the model to effectively learn and retain patterns over long periods when processing sequential data. Experimentally, it has been shown that the number of layers in a given LSTM model plays a crucial role in enhancing the performance of sequential data analysis [19]. In a restricted environment, as the number of layers in a network increases, its ability to discern intricate patterns and representations from the data grows. Deeper LSTM networks (Deep LSTM) can decipher hierarchical characteristics and relationships within the data, empowering them to represent more subtle connections and possibly excel in intricate tasks [20]. Therefore, the proposed research prefers Deep LSTM over other competitive models for significant feature extraction from sentiment data samples. The proposed deep LSTM was utilized to collect features and dependencies present in the input texts.

Further, a hybrid feature reduction method, based on PCA(Principal Component Analysis) and SVD (Singular Value Decomposition) was implemented to reduce the high dimensionality of generated features [21]. PCA is a popular feature reduction technique used for mapping original features with lower dimensional features with similar

semantic data. It has also been used for converting sparse vectors to a dense level of embedding vectors. SVD is another significant approach based on linear algebra technique with diverse applications in machine learning. It enables tasks like dimensionality reduction, data compression, noise reduction, feature extraction, and latent factor analysis. Previously, several baseline studies have shown the significance of hybrid feature reduction techniques for the effective prediction of emotions from text data. Finally, the optimized feature set was fed into the Random Forest (RF) classifier [22] to discriminate happy and sad emotions and global performance was reported. Random forests can learn various non-linear relationships between reduced features and target variables supplemented with the interaction between the features and, as a result [23]. Another convenience of random forests is their inbuilt feature importance estimating tools which help practitioners effectively isolate the most effective features of the classification process. Also, a five-fold cross-validation [24] approach was employed to quantify classification results statistically to split the global population into training and evaluation sets.

The primary objectives of the proposed research study can be recapitulated as follows:

a) To retrieve human emotions after reading from stories using semantic analysis with the integration of Deep feature extraction and dimensionality reduction with an updated classification model.
b) To accomplish the feature extraction task with DL-based LSTM Layer for obtaining the significant and relevant features from the feature set.
c) To perform dimensionality reduction operation with Bi-orthogonalization using PCA and SVD to solve issues of scalability and sparsity.
d) To perform classification with weight updated tuned RF classifier with weights from the LSTM layer for segregating emotions based on stories and to enhance the accuracy rate in classification.
e) To evaluate the performance of the system with evaluation metrics such as precision, accuracy, recall, cappa score, and F1 score.

The remaining structure of the paper is as follows: Section II recapitulated the baseline research studies in the semantic analysis. In Section III, the proposed improvement strategy is explained. The experimental results and discussion are provided in Section IV. Finally, conclusions and the future scope of the work are presented in Section V.

II. RELATED WORKS

NLP is crucial for sentiment analysis, providing machines to interpret human language and obtain sentiments from textual data through techniques such as feature extraction and preprocessing. ML algorithms such as SVM, Naive Bayes, and Logistic Regression are employed for the classification of sentiment which depends on the manual feature engineering [25]. Besides, DL methods such as RNNs, CNNs, and transformers have proven to be effective in this field, automatically obtaining patterns from raw text without any feature engineering and thus able to capture sentiment features at both local and contextual levels [26]. The two significant methods utilized for topic modeling were PLSA (Probabilistic Latent Semantic Analysis) and LDA [27]. It consists of a group of discrete information of a three-tier hierarchical Bayesian framework in which every time a collection has been modeled with the finite mixture of underlying groups of topics. It has been considered the unsupervised technique in data mining through which concealed topics of the document have been derived from the estimated probability distributions while neglecting the order of words. The model has been designed in such a way that the underlying topics related to the document are in the range of multi-nominal distribution of words within the fixed context. LDA-based modeling has provided its advantages over numerous applications like marketing, education, management, and social networks [28]. Analyzing emotions through words has helped numerous businesses extract their clients' needs. The findings from the considered methodology have verified improvement in service through sentiment analysis and have indicated significant topics that have concentrated on enhancing the quality of service in firms through analysis from various sources. The Naïve Bayes technique in the recommended research has been utilized for the classification of text based on rougher assumptions with easier and faster implantation [29]. The simpler and heuristic solutions for some problems have been identified with the MNB (Multinomial Naïve Bayes). It is considered to be a kind of NB classifier utilized for SA. The suggested methodology [30] has utilized a dataset for movie reviews, and every review has consisted of a notice comprised of text and number-based scores ranging from 0 to 100. The widely used reference datasets utilized for classification have confirmed the effectiveness of the algorithm since accuracy has been attained at a greater level with MNB classifiers.

The research with SA has gained numerous advantages with practical applications. The preferences made by users have been evaluated, and based on that, recommendations have been provided with a specific target, which increases sales [31]. Many conventional analysis methods have been utilized, such as KNN (K-Neraest Neighbour), SVM (Support Vector Machine), and CART (Classification and Regression Tree), but have achieved poor levels of accuracy. The suggested technique [32] has provided enhanced K-NN-based classification in which the normalization of data has improved the metrics of accuracy and precision. The THUCNews dataset [33] has been used for the validation of the methodology. It has reduced the running time and enhanced the sentiment-based classification. The SA is designed for the classification of documents into a fixed group of previously defined categories, which have been represented with different sentiments.

Due to the complications of inadequate training in data faced in earlier research, the recommended study [34] has performed a multi-task-based learning framework based on

DL (Deep Learning) and has achieved progress significantly. Generally, models balance various datasets interpreted for multiple tasks to improve the performance of individual datasets. The enhancement has been evident in jobs with a limited amount of training in the data. Many frameworks have suffered from several limitations. Initially, the traditional approaches have utilized the outcome of the hidden layer as the representation of text, which has lost some amount of information [35]. Following that, even though algorithms have utilized the gating technique for selecting some shared features, certain irrelevant features have erroneously occurred. The suggested methodology [36] has provided solutions for irrelevant information with a pooling layer integrated into the BRNN (Bi-directional Recurrent Neural Network) for the comprehensive extraction of semantic information. The attention mechanism between the task-oriented layers and the shared layers has been applied for the identification of the features. Attention-based separate pooling technique has been conducted for the effectiveness of the model with different datasets. The effectiveness of the algorithm has increased steadily with the addition of components [37]. Further evaluation of the datasets has extracted useful information, and the link of specific layers with the shared layers has empowered the framework for selecting effective features from shared layers. Such features have allowed for conducting the task of efficient SA specifically to smaller datasets with the utilization of different tasks efficiently. Several applications have been implemented for the SA process, like prediction of markets, review of products, determination of political sentiment, and prediction of equivalence value [38]. Numerous types of research have been utilized for the prediction of positive and negative sentiment depending on words and methods have experienced limitations with the predictable sentiment and multiple opinions from various users [39]. The intention of SA lies in the extraction of objectivity of opinions. The normal tweet is comprised of different features other than general text. From the time of the tweet, identifying the number of tweet texts, and the location of the text posted have been noted, and performed an analysis of certain attributes that assist in the improvement of classification [40]. The suggested technique [41] used the blogging application Twitter since people share different opinions on particular topics in that blog. It has utilized like and retweet techniques for improving SA. The prediction of features has been performed through Naïve Bayes to classify text and non-textual features through the Fisher score technique. The accuracy values have been obtained optimally through effective techniques. The comparison of various methods is tabulated in Table 1.

TABLE 1. Recapitulation of the baseline methods used in SA.

Ref.	Methods	Outcomes
[19]	Consumer reviews analysis by topic modeling approach with CNN and LSTM.	The ensemble model outperforms other machine learning approaches in terms of accuracy and other parameters.
[42]	Prediction and analysis of the Indonesia Presidential election from Twitter using sentiment analysis with an algorithm based on R language	With the top keywords, reliable prediction results on elections.
[31]	An ensemble approach to stabilize the features for multi-domain sentiment analysis using supervised machine learning like RF, LR, MNB, and SMO	Sentiment classification with 92% accuracy.
[32]	An improved text classification method for sentiment classification using enhanced K-NN	Improved K-NN achieved an accuracy rate of 11.5% in performing classification.
[34]	Recurrent neural network with pooling operation and attention mechanism for sentiment analysis	Accuracy is enhanced by 0.5 % for every model
[35]	Attention-based long short-term memory network using sentiment lexicon embedding for aspect-level sentiment analysis in Korean	Achieved time consumption and improved performance.

III. RESEARCH METHODOLOGY

A. DATA ACQUISITION

The present study performs sentimental analysis using a 4000 children's stories dataset. The 4000 short stories in the dataset [43] were categorized based on their emotional content and semantic structure. This real-time dataset was produced for the sentiment analysis research. Several fields make up the dataset: URL of the story, valence, arousal, dominance, length, title, story name, valence, arousal, dominance, text number, author of the story, story name, word_count, sentence_length, text identification, processed_text, and other feature vector attributes. The doc2vec technique was used to create a semantic framework in which each text is classified as a 300-place vector. The author created the dataset as part of a study on the impact of narrative literature on mental health. This dataset is publicly available and can be accessed with the source repository [44].

B. TEXT PRE-PROCESSING

Initially, the existing children's stories dataset was loaded and several pre-processing steps were applied to enhance the quality of the input text. In the first step, the missing values and the unstandardized feature values are processed with proper scaling values before feature extraction. To do this, the Word embeddings [45] approach was implemented to identify semantic links between words in a vector space. Here, we assumed that similar words typically have similar embeddings. In this context, a Cosine Similarity [46] measure was used to determine the accurate alternative of the missing word. Further, the dataset was preprocessed to remove the noise and prepare the data for classification by using one of the most popular NLP libraries, NLTK (Natural Language Toolkit) [47], which includes packages to help machines comprehend human language and respond to users in the desired manner. Here, three substeps: (1) Data

TABLE 2. The statistical details of parameters used in the Decay Schedule algorithm.

Parameter	α_{old}	Factor	Epoch	Step_size
Values	0.9	0.001	variable	0.1

tokenization, (2) Text cleaning (links, emojis, and special characters removal), and (3) Data labeling using the Python Vader (Valence Aware Dictionary and Sentiment Reasoner) tool [48]. Terms are generally categorized as positive or negative based on their semantic orientation in this rule-based sentiment analyzer. In this study, a happy story is defined as having a polarity score greater than 0.5; if the polarity score is significantly lower than 0.5, it is regarded as a negative (sad story).

C. DEEP LSTM FOR FEATURE EXTRACTION FOR IMPROVISING LEARNING RATE

Deep LSTM is one of the extensions of RNN, which has multiple LSTM layers. RNN is a special category of supervised DL algorithms widely used in applications like speech recognition, natural language processing, and image captioning [49]. Here, our proposed Deep LSTM model consists of 8 hidden and 2 dense layers. The proposed architecture is shown in Figure 1. Each hidden layer is composed of four gates: (1) Forget gate, (2) Input gate, (3) Output gate, and (4) Cell state. The forget gate controls the level of information from the previous time step that is retained in the current time step. The input gate controls how much new information can be added to the cell state in the current time step. The output gate controls how much of the cell state is output from the LSTM layer in the current time step. The cell state is a memory cell that stores information over time. A set of weights and biases controls these gates. These weights and biases are learned during the model training process. The architecture of a deep LSTM is similar to that of a single LSTM layer, but multiple LSTM layers are stacked on top of each other. The output from each LSTM layer is fed into the following LSTM layer. This allows the deep LSTM to learn long-term dependencies present in the data. Depending on the task, the number of LSTM layers in a deep LSTM can vary.

During the learning process, it was required to adjust a suitable learning rate to minimize the loss and maximize the relevance of the extracted features. The learning rate is a hyperparameter, that controls the weight ranges of a machine-learning model and is automatically updated during model training. A higher learning rate can result in the model to converge too quickly. Similarly, a lower learning rate can result in slower model convergence. In the proposed study, we employed the ''Decay Schedule'' [50], a popular scheme used to systematically reduce the learning rate over the training period in a machine learning model. The ''Decay Schedule'', policy is described in Eq. 1. The learning process was performed in Algorithm 1 for 250 epochs. The details of all the parameters are discussed in Table 2.

$$\alpha_{updated} \rightarrow \alpha_{old} * factor^{floor(\frac{epoch}{step_size})} \tag{1}$$

Algorithm 1 Deep Feature Extraction Using LSTM Layer

Input	*Data from preprocessing*
Output	*feature extracted using* $D-LSTM$ $y = [y_1, \ldots..y_N], y_i P^n$ *Total number of hidden layers* $= 8$, *dense layers* $= 2$ *params*: $M_f, N_f, b_f -$ *Forget Gate*, $M_{\tilde{C}}, N_{\tilde{C}}, b_{\tilde{C}}, -$*Candidate Gate*, $M_i, N_i, b_i -$ *Input Gate* $M_o, N_o, b_o -$ *Output Gate*
Initialization	Set initial cell state c_0 and hidden state z_0 to zero vectors
Forward Pass	Forward Pass: for t = 1 to 365: - Calculate gate activations: - g_t = sigmoid(M_f * X[t] + N_f * c_0 + b_f) - i_t = sigmoid(M_i * X[t] + N_i * c_0 + b_i) - o_t = sigmoid(M_o * X[t] + N_o * c_0 + b_o) - Calculate candidate update: - c_t = g_t * tanh(M_c * X[t] + N_c * c_0 + b_c) - Update cell state: - c_0 = c_t - Update hidden state: - z_t = o_t * tanh(c_t) Store hidden state z_t for each time step t
Output Calculation	Calculate output as a dot product of weights with hidden states z_t
Final Output	Return the final output feature vector

- α_{old} : The initial learning rate.
- factor: The factor by which the rate is reduced.
- epoch: The current epoch number
- step size: The interval at which the learning rate is reduced.

In Figure 2, we have shown the K-mean distribution of extracted features from both Happy (Orange) and Sad sentiments (Blue), respectively. The circular marked region showed overlapped features with a lesser discriminable nature during classification. The extracted features were further fed into a hybridized bi-orthogonal SVD and PCA modules to eliminate irrelevant features. In other words, our target was to minimize the feature overlapping region so that the selected features could be easily discriminated by the Random Forest (RF) classifier. In Figure 1, the initial layer (embedding_1) is the first one to deal with this conversion of input integer sequences into dense vectors of a predefined size. Notably, it converts variable length inputs to dense vectors of length 128 and this can accommodate the varying lengths of inputs to form consistent patterns. This is the essential meaning of this layer that contributes towards sequence processing fast even at the stage of large unknown sizes. Mention that each LSTM layer (lstm_1 to lstm_8) is processing the information from the output of either the previous layer or the embedding layer. Conventional RNNs suffer from ''vanishing gradients'' issues as a rule. LSTMs overcome this barrier and make themselves excellent tools for incorporating long-term dependencies into sequential data. The proposed deep learning architecture is given in Table 3.

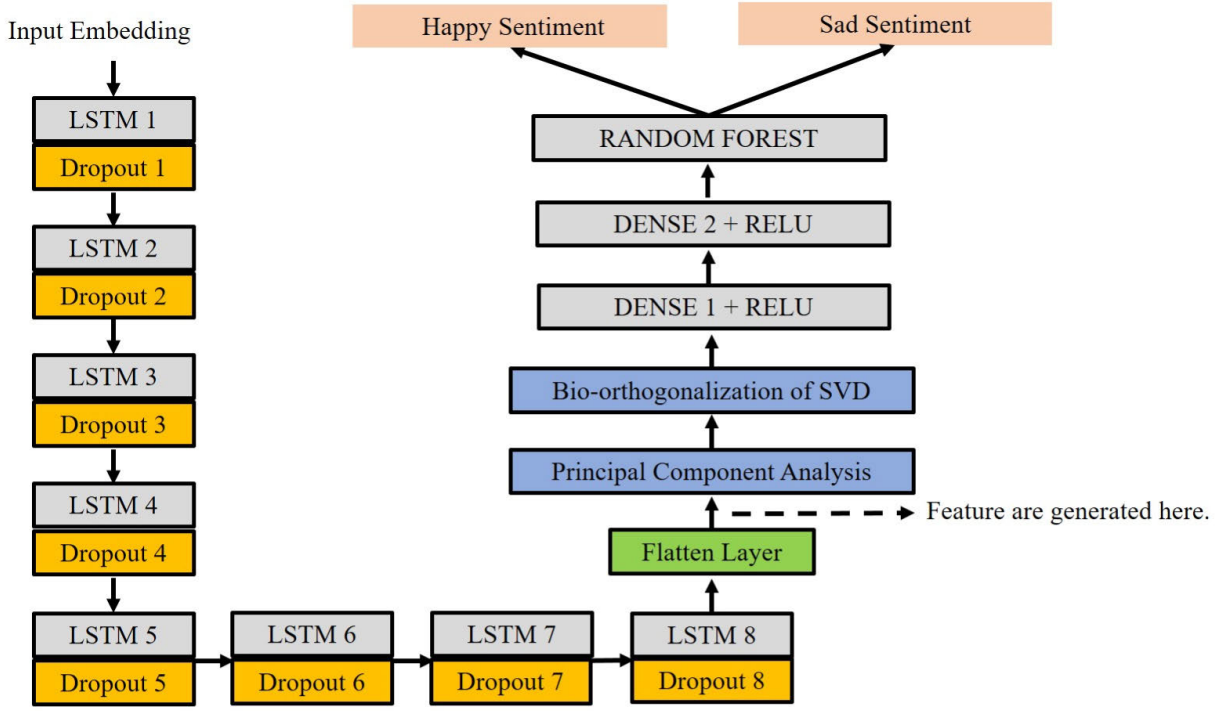


FIGURE 1. Proposed classification architecture using 8 LSTM layers, 1 flattened layer, Principal component analysis, SVD Bio-orthogonalization, and Dense layer.

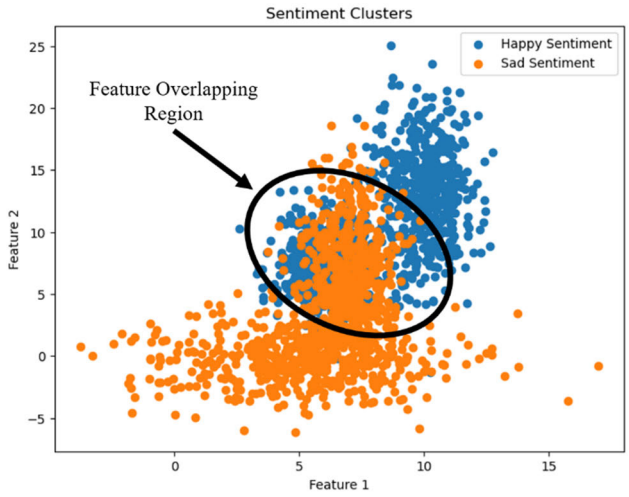


FIGURE 2. Visualization of features extracted from "Happy" (Blue) and "Sad" (Orange) sentiments. Here, the feature overlapping region shows the number of features from both classes with minimum discriminable ability.

The output shape values for every LSTM layer are (None, 20, 64), where 'None' corresponds to the batch size, '20' stands for the sequence length (20 in this example), and '64' indicates how many units are in each LSTM cell. Importantly, each LSTM layer is made up of 49,408 trainable parameters, which can be split into weight (autonomous) and bias (additional) values during training. After the flattening curves the output to a one-dimensional array for the upcoming Layers (dense_1 and dense_2). Along the Denser Adjacencies, occur

TABLE 3. The architecture of the deep LSTM architecture.

Model: "sequential_2"		
Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, None, 128)	2560000
lstm_1 (LSTM)	(None, 20, 64)	49408
lstm_2 (LSTM)	(None, 20, 64)	33024
lstm_3 (LSTM)	(None, 20, 64)	33024
lstm_4 (LSTM)	(None, 20, 64)	33024
lstm_5 (LSTM)	(None, 20, 64)	33024
lstm_6 (LSTM)	(None, 20, 64)	33024
lstm_7 (LSTM)	(None, 20, 64)	33024
lstm_8 (LSTM)	(None, 20, 64)	33024
flatten_2 (Flatten)	(None, 1280)	0
dense_1 (Dense)	(None, 64)	81984
dense_2 (Dense)	(None, 1)	2080
Total params:	$2.9246*10^6$	

the classifications whose roles are only represented by the presentation of the LSTM features. The final shape for the dense_1 block is (None, 64), and for the dense_2 is (None, 1), suitable for making binary classification. Sparsity is defined as dense_1 parameter 81984 and in dense_2 are 2080. So basically we got 2.9 million parameters in a model overall, and these trainable ones are going to be updated during training and that's exactly what will happen in our LSTM and dense layers.

D. HYPERPARAMETER TUNING

Hyperparameter tuning is a crucial step to success during the deployment of the LSTM model for performing classification. Among a multitude of methods for this task, Bayesian optimization is a very effective tool. Bayesian

optimization [51] efficiently traverses the hyperparameter space and smartly chooses the optimal combination by considering a known objective function. This strategy maintains a proper balance between exploration and exploitation, which enables it to readily find feasible spaces with high values of the performance metric without using a lot of computational resources. The pseudocode of customized Bayesian optimization for LSTM parameters is given in Algorithm 2.

Algorithm 2 Bayesian Optimization for LSTM Parameter Tuning

Input	- Search space X for LSTM hyperparameters: - Number of LSTM Units per Layer: 50 to 500 units per layer - Learning Rate: 0.0001 to 0.01 - Dropout Rate: 0.1 to 0.5 - Recurrent Dropout Rate: 0.1 to 0.5 - Number of Epochs: 200 epochs - Batch Size: 16, 32, 64, 128 - Optimizer: Adam - Activation Function: ReLU - Loss Function: Categorical Crossentropy, Mean Squared Error - Early Stopping
Output:	Optimal set of LSTM hyperparameters x∗ with balances accuracy and minimum training time
Initialize	- Select initial sample points x_1, x_2, …, x_n from the search space X - Evaluate objective functions with high balances accuracy and minimum training time
Build Initial Model	Build surrogate Gaussian probabilistic models to approximate the objective functions: (1) Balances accuracy and (2) Training time based on the evaluated sample points
	Iterate until the stopping criterion is met: a. Select the next sample point: - Use the acquisition function $\alpha(x)$ to determine the next sample point that balances exploration and exploitation while considering both accuracy and training time objectives b. Evaluate objective functions: - Evaluate the objective functions: (1) accuracy and (2) training time for the selected sample point c. Update surrogate models: - Update the surrogate models with the new observations of accuracy and minimum training time d. Update acquisition function: - Update the acquisition function based on the updated surrogate models e. Repeat steps a-d until the maximum number of iterations are reached
Output	Return the optimal set of LSTM hyperparameters x∗ that maximizes accuracy while minimizing training time based on the surrogate models

E. DIMENSION REDUCTION

The extracted features ought to be reduced in dimension, which can be accomplished using PCA (Principal Component Analysis) and bio-orthogonalization SVD (Singular Value Decomposition) methods. The features retrieved from the Deep LSTM model are passed together as a single linear set of features to enhance PCA and SVD for minimizing the dimensions of features. Certain dimensions contain noisy and irrelevant data, whereas others capture semantic relationships or patterns in data. To handle such complications, PCA assists in identifying the highly significant dimensions in input data and reduces the data to a smaller number of principal components [21]. The resulting principal components are input to the ML model. Several other benefits of using PCA in SA included eliminating noise and grasping irrelevant information, which improves accuracy considerably. The linear set of features was again reduced in low dimensional feature space through the SVD algorithm. Once linear features are chosen, classification occurs. Most classes from each decision tree in the random forest classifier are classified.

The weights of variables incorrectly predicted by any decision tree increased and were fed to the next decision tree. The weights assigned by LSTM layers to every feature were updated to the feature class of RF classifier tree values. Hence, the parameter tuning is accomplished through the LSTM neural network. SVD improves the model's capability with a new set of classified linear features and improves the prediction accuracy of sentiments. The performance of the model is analyzed through comparative analysis and metrics.

F. DIMENSIONALITY REDUCTION USING BIO-ORTHOGONALIZATION PCA

The Principal component analysis (PCA) is a linear dimensionality-reduction technique, which converts high-dimensional data to low-dimensional data by increasing the lower-dimension variance [21]. This sort of transformation can be arranged in that the first PC possesses the largest variance and every proceeding component also possesses a larger variance under orthogonal conditions than other succeeding components. The feature vector co-variance matrix is first computed and proceeded by eigenvector computation of the matrix. Those eigenvectors which possess the largest eigenvalues result in the formation of new feature vectors having minimal dimensionality. Hence, rather than losing a few significant data features, the PCA algorithm aids in keeping out the most significant features by retaining 99 percent of the variance. Before this, dimensionality reduction, pre-processing, and extraction of significant features are obtained. The mean of each feature has been computed below:

$$\rho_i = \frac{1}{n}\sum_{j=1}^{n} Zz_i^{(j)} \tag{2}$$

Here, z signifies the training set, ρ_i is the mean of each feature, and n represents the dimensional space for the training set.

Therefore, every z_i value with z_i- ρ_I value is been replaced; hence, each of them had zero for mean value, but also, if various features had different mean values, one had to

do feature scaling such that must be compared to a specific comparable level. In that supervised learning, the process of scaling of i^{th} has been defined by below equation, wherein s_i is defined as $|max - mean|$ or the feature's standard deviation of i^{th} feature:

$$\left(z_i^j\right) = \frac{z_i^{(j)} - \rho_i}{s_i} \tag{3}$$

To minimize the dimension, from N to m, and to define the surface in N-dimensional space onto which data has been projected, it is necessary to determine the mean square error of such projected data on the m-dimensional vector. Otherside, the eigenvectors would have smaller wigen-values, having small dataset information:

$$\text{Covariance matrix} = \frac{1}{N}\sum_{j=1}^{n} Pz^{(j)} \times (z^{(j)})^T \tag{4}$$

Therefore the p^{th} principal component of data-vector $z^{(j)}$ in those transformed co-ordinates could be assigned $t^p = z^{(j)} \times w^{(p)}$ wherein this $w^{(p)}$ represents the $z^{(i)}\text{co} - \text{variance matrix}p^{th}$ eigen vector. After that, the threshold value beyond which the entire useful eigenvalues are considered was been set, and the rest features were discarded to be non-significant features. The threshold value is set to 99 because it helps to minimize the amount of features without losing maximum information. Besides, it enhances the interpretability of the system by minimizing the amount of features. Moreover, 99% variance prevents the system from overfitting data by reducing the features. Corresponding, the 99% variance states that 99 percentage of data variance is retained within the minimized feature vector. As LSTM extracts various having distinct modalities, the count of extracted features after PCA seems different based on feature set distribution.

1) SVD FOR DIMENSIONALITY REDUCTION WITH HANDLING SPARSITY ISSUES

To provide better outcomes after reducing dimensions, SVD is being incorporated along with PCA. This SVD is another efficient method to reduce the high dimensions of the feature vector [21]. To determine the low-dimensional feature space and its scalability, sparsity issues stand as a prominent challenge.

The SVD of a matrix having size a × b is obtained:

$$svd\ (T) = GIH^t \tag{5}$$

where G and H were the a × a and b × b orthogonal matrix. This 'I' represents a × b singular-orthogonal matrix having non-negative elements. The SVD operation is processed after the PCA process. An a × a matrix G is defined as orthogonal if this G^tG is equal to the identified matrix having a × a size. The diagonal elements within $\Sigma(\sigma 1, \sigma 2 \ldots \sigma n)$ were the singular matrix values. Generally, singular values were placed in descending sorting order in I. The right and left vectors were the column vectors of B and A. Many significant applications utilize desirable properties of SVD. The truncated-SVD having rank z has been defined by:

$$svd\ (T_z) = G_z I_z \tag{6}$$

In equation 5, G_z and H_z were the a×z and b×z matrices, comprised of the first z columns within A matrix and the first similar z columns within matrix B. The matrix I_z represents the z×z principle diagonal sub-matrix of I. Similarly, T_z denotes the closest original matrix linear approximations. Once the feature transformation is accomplished, the items and features could be thought of be feature points within z-dimensional space. The steps involved in the process of PCA and SVD are signified below: The PCA is used in the system to minimize the input data's dimensionality. However, it recollects the variance in the data without losing extra information. The dimension in the input data is reduced because only a limited dimension in the data is significant for the classification. Moreover, certain dimension in the data contains noise and unwanted information. For this purpose, the PCA is used for dimension reduction in the proposed system. The steps involved in the main components are the linear combination of the input data. It is selected to maximize the data variance, although reducing the information loss in the input data. Correspondingly, the input of PCA and SVD is defined as the matrix of data. Here, the rows and columns of the matrix signify the data points and features. The working of SVD is discussed the Algorithm 3.

Algorithm 3 Definitive Bio-Orthogonalization of SVD

Input:*Extracted feature from the lstm were mat* $(a \times b)\ T$, *and a* $num'z'$
Output:G_z, H_z, I_z
begin
initiate_svd()
Generate an $b \times z$ *Gaussian matrix of GM*
$Y = TGM$
Calculating an orthogonal col_{basis} *P of Y*
$B = P^t T$
compute $eign_{decomposition}(BB^t) = CI^2C^t$
$G_z = PI, H_z^t = B^t CI^{-1}, I_z = I$
end

Initially, in the dimension reduction, the covariance matrix is calculated. It measures the similarity among the features. Then, the covariance among the two features is computed to take the average of the deviation in the product. It is processed by the mean value of each feature. Besides, the eigenvalues and eigenvectors of the matrix are calculated. It is the vectors and values which diagonalize the covariance matrix. Further, the desired value for the principle components is selected in the system. Finally, the data is estimated in the principle components by increasing the data matrix through the eigenvector matrix. It will be the outcome in the new matrix, which is the minimized space dimension. Formerly, the efficiency of the dimension reduction is calculated by the loss of information between the projected data and the original data. The final projection of the feature distribution of ''Happy'' and ''Sad'' sentiments is shown in Figure 3.

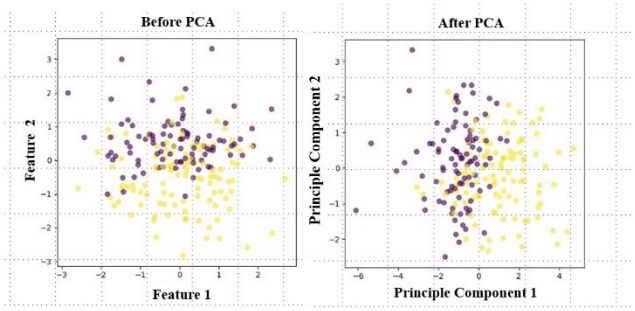


FIGURE 3. Visualization of before and after PCA implementation over-extracted deep LSTM features.

G. RANDOM FOREST CLASSIFIER –HYPERPARAMETER TUNING WITH LSTM WEIGHTS UPDATE FOR CLASSIFICATION

Random Forest (RF) is a kind of ensemble learning algorithm utilized for effective classification and regression since it makes predictions from multiple trees and combines them to make a final prediction [52]. It has the potential to handle both continuous and categorical data and capture complex and non-linear relationships between attributes. Moreover, tuning with RF enhances the accuracy rate, especially for longer text- sequences. One of the key advantages of using RF with Deep LSTM relies on the fact that this method addresses the vanishing gradient issue, which makes the model training more difficult for long sequences of integers or words. After features are reduced, an RF (Random Forest) classifier is used for classifying the sentiment on text based on reduced features. Random forests are constructed on decision trees to be the atomic units. Every decision tree behaves as a classifier for output prediction. Once the features are extracted, the network model classifies the features to be happy or sad expressions. Tuning in the classifier optimizes the hyperparameter such as maximum depth, minimum samples required for splitting the node, and number of trees. Algorithm 4 depicts the in-depth function of the RF classification in the respective research.

The weight transfer from the LSTM to RF enhances the amount of tress in the RF which improves the performance of classification. The decision tree algorithm responds to classification tasks through tree representation. The labels of the class were assigned to every leaf node. The significant accuracy improvement in feature classification to the category of emotion features had led to a growing ensemble of trees and let them go for the most probable or popular class values assigned with the highest LSTM weights through searching out the decision tree and choosing the best tree. To grow those ensembles, random vectors were generated governing every tree's growth in the ensemble. This random forest classifier classifies the expression features using tree-structured classifiers. The data TS is trained with an alternate for generating TS_i. The base root node is created RN_i for that TS_i. The tree is then built by forming closeness to accurate value among the features that aid the classification process. If the root node possesses 0 class of features, it is classified as a negative value. Conversely, if the root node possesses a '1' class, the positive classification occurs. The tree is then split into several c child nodes, wherein C possesses values ranging from C_1,…C_f. All the contents of RN_i were set towards all instances of RN which has similarity/match C_i having weight value sample_weight. This is the weight value of features gained from Deep LSTM feature learning. The hyperparameters were also tuned to bring out enhancement in classification. With such a weighted sampling feature strategy, the final results will be capable of focusing on the most significant hyperparameters while not letting out any features.

Algorithm 4 Random Forest Classification

Begin
init a classifier
for i =1 to a do
train data TS with alternate to produce (TS_i)
form a root node, RN_i (TS_i)
Call BuildTree (RN_i)
end for
BuildTree(RN):
if RN has samples of only 0 class then
return(positive)
If else if RN has samples of only 1 class, then
return(negative)
rand (x) split features in RN
Select the Feat with the highest info gain to split on
Create c child nodes RN_1, RN_2, ……..RN_f, where C has c possible values (C_1, …..C_f)
for i =1 to c do
set the contents of RN_i to TS_i, where TS_i is all instances in RN that match
$C_i$with $samplee_{weight}$= weightLSTM [0] [0] [i]
Call BuildTree with weight(lstm) (RN_i−lstm(w))
Return (high accuracy with weight)
end for
end if

IV. RESULTS AND DISCUSSION

The following section illustrates the performance analysis of the proposed expression classification in the sentimental analysis regarding different metrics and explicates the comparative analysis section. The above-discussed experiment setup was validated by decomposing the original dataset into five equal groups using a five-fold cross-validation technique. The model is trained and evaluated five times, each time using a different fold as the validation set and average accuracy scores were reported. SVD and PCA with RF methods provide an effective solution for sentiment analysis by combining math techniques and machine learning to simplify the data and make accurate predictions. Further, a chi-squared test was applied to show the significance of the extracted features and the top 10 features were plotted in decreasing order of score (Figure 4). The selected features were classified using RF classifiers and 99.19% classification accuracy was reported.

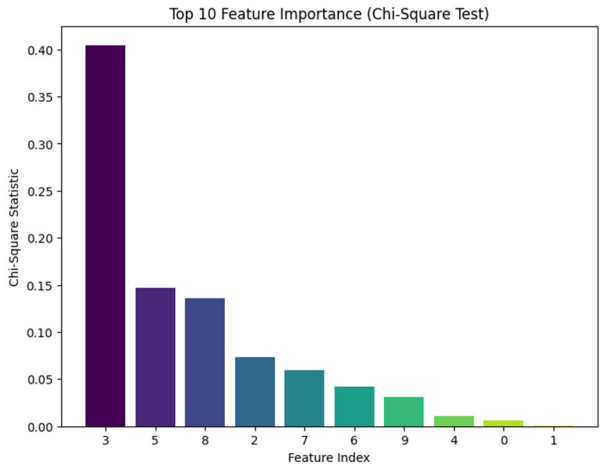


FIGURE 4. Chi-squared statistic of top 10 features in the decreasing order.

The algorithm of the Random Forest has several hyperparameters that can be adjusted for detecting optimal performance. The metric of n_estimators sets the number of trees within the forest; with a high value, the performance may improve, however, the computation complexity might also increase. Initially, we set 100 trees as our first value. The max_depth variable controls the maximal length of each tree, with more complex patterns detected by deep trees and the downside of overfitting associated with them. To mitigate overfitting, we adjusted the max_depth parameter equal to 10. On the other hand, for min_samples_split, the key is to find the right balance, which determines the minimum number of samples needed to split an internal node. A value set too high might induce underfitting while a value set too low will possibly result in overfitting. Here, we initialized this variable with the value of 5.

Likewise, min_samples_leaf provides a chance to choose the minimum number of samples that should be at a leaf node acting as a regularization parameter which prevents the model from overfitting. A value of 2 is set to begin the experiment. The max_features parameter defines the maximum number of attributes that are selected for the best-split search. The number of attributes considered then defines the randomness of the trees and avoids overfitting. Initially, values such as 'sqrt' (square root of the dimension of the feature space) or 'log2' (log base 2 of the feature space dimension) are usually chosen. Besides that, the bootstrap parameter allows the sampling of bootstrap to create trees, which makes the decision randomly to improve the generalization. Following the conventional training process, it is set up to True as bootstrap sampling is employed to generalize the model and minimize the risk of overfitting. The details of optimized hyperparameters (responsible for maximum classification accuracy) are reported in Table 4.

In Figure 5, we have shown the epochwise performance of the proposed classification framework. The performance curve clearly illustrates that the proposed model effectively deals with the overfitting issue because of a suitable

TABLE 4. Initial parameter selection for random forest classifier.

Parameter	Description	Value
n_estimators	Number of trees in the forest	980
max_depth	Maximum depth of each tree	48
min_samples_split	The minimum number of samples required to split a node	2
min_samples_leaf	The minimum number of samples required to be a leaf node	1
max_features	Maximum number of features considered for splitting	'sqrt'
bootstrap	Whether bootstrap samples are used when building trees	True
class_weight	Weights associated with classes in the form {class_label: weight}	'balanced'

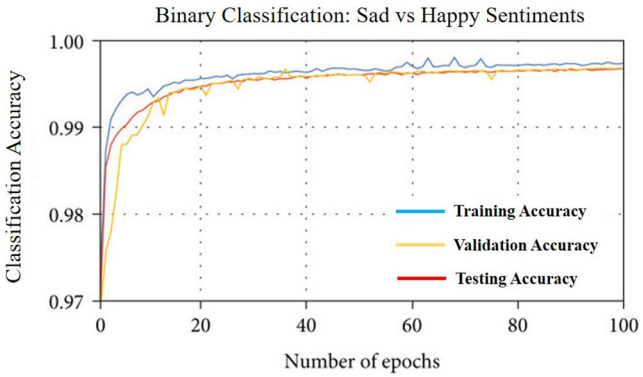


FIGURE 5. A visualization of the epochwise performance of the proposed classification framework.

hyperparameter tuning procedure. Moreover, It can be easily observed that the proposed model achieved steady classification accuracy scores after 60 iterations. In addition, the average classification accuracy of 99.19% was reported which was superior to five baseline classification approaches: (1) Naïve Bayes (NB), (2) Gradient Boosting Machines (GBM), (3) Support Vector Machine (SVM), (4) Vanilla LSTM (VLSTM), and (5) Ada Boost Algorithm (ABA). To enhance the scope of the proposed research work, we used two different types of features with baseline classification techniques. The first type of feature is the sentiment feature which consists of a combination of the sentiment, the strength of the sentiment, and the polarity of the sentiment from a sentiment dictionary if that's available. The dictionaries used for the task are the Dalian University of Technology Sentiment Dictionary (DUTSD) [53], and the National Taiwan University Sentiment Dictionary (NTUSD) [54]. The problem with these dictionaries is that they do not encompass all the sentiment words, therefore, raw segmented words are also used as the second form of feature. The feature extraction is done using a common bag of word representation. The first dictionary offers a variety of features as a collection of words and phrases with assigned sentiment scores for sentiment analysis. On the contrary, the second class refers to a

TABLE 5. Details of features extracted from both dictionaries (DUTSD and NTUSD).

Sr. No.	Dalian University of Technology Sentiment Dictionary (DUTSD)	National Taiwan University Sentiment Dictionary (NTUSD)
1	Sentiment Scores	Sentiment Scores
2	Word Frequency	Word Frequency
3	Sentiment Distribution	Sentiment Distribution
4	Sentiment Density	Sentiment Density
5	Sentiment Variation	Contextual Analysis
6	Sentiment Trends	Sentiment Trends
7	Contextual Analysis	Sentiment Correlation
8	Sentiment Aggregation	Sentiment Aggregation
9	Sentiment Classification	Sentiment Classification
10	Sentiment Correlation	Emotion Analysis
11	Sentiment Intensity	-
12	Sentiment Bias	-
13	Sentiment Emotion Analysis	-
14	Sentiment Subjectivity	-
15	Sentiment Contrast Analysis	-

sentiment dictionary based on the lexicon (words) developed by the researchers at National Taiwan University. It stands for the compilation of such words or phrases with their sentiment range scores which come on the positive, negative, or neutral sides. In our work, we ignored neutral features because of the binary scope of our classification problem. The details of features extracted from both categories of feature extraction techniques (Dictionaries) are given in Table 5.

The realized outcomes were compared with state-of-the-art methods using six performance metrics: (1) Classification accuracy, (2) Precision, (3) Recall, (4) F1-score, (5) Cohen's Kappa statistic, and (6) Time complexity. The first measure, classification accuracy refers to the ratio of correctly classified observations out of the total observations in the dataset (Equation 7).

The second score indicates the ratio of correctly predicted positive samples out of all samples predicted as positive (Equation 8). Recall is another metric that shows the ratio of correctly predicted positive observations out of all actual positive observations (Equation 9). The harmonic mean of recall and precision is the F1 score (Equation 10). It is especially helpful for datasets that are imbalanced since it strikes a balance between recall and precision. Cohen's Kappa statistic calculates the degree of agreement between the actual and anticipated labels while accounting for the possibility of random error (Equation 11). It is especially helpful for assessing how well sentiment classification frameworks function when there is a class disparity. Finally, time complexity calculates how long the algorithm needs, given a particular input size, to process and evaluate textual material to forecast sentiment.

$$Classifcation\ Accuracy = \frac{Correctly\ predicted\ instances}{Total\ instances} * 100 \tag{7}$$

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{8}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{9}$$

$$F1 - Score = 2 * \frac{Precision * Recall}{Precison + Recall} \tag{10}$$

$$\text{Cohen's cappa value} = \frac{p\,(A) - P(B)}{1 - P(E)} \tag{11}$$

In each of these formulas:

- True Positives: The number of instances that are correctly predicted as positive.
- False Positives: The number of instances that are incorrectly predicted as positive when they are negative.
- False Negatives: The number of instances that are incorrectly predicted as negative when they are positive.
- Precision: The proportion of correctly predicted positive instances out of all instances predicted as positive.
- Recall: The proportion of correctly predicted positive instances out of all actual positive instances.
- $P(A)$: The observed agreement between predicted and actual labels.
- $P(E)$: The expected agreement under random chance.

The detailed comparative results between the proposed classification model and state-of-the-art methods are given in Table 6. It can be observed that the proposed approach (Deep LSTM feature + RF) achieved a maximum classification accuracy of 99.19% which is superior to other competitive strategies. It is interesting to discuss that our induced hybridization procedure significantly boosts the classification performance (4.6%) compared to the original LSTM method (without any optimization). It showed notable compatibility between RF and deep LSTM features and their ability to discriminate complex text patterns. In this case, the sequence of all the performing methods is LSTM+ RF > VLSTM > SVM > GBM > NB> ABA. In addition, the global results indicate avoiding ABA in text classification because this method has been found to suffer from the problem of overfitting in some instances during training. Similarly, the proposed approach achieved high precision and recall scores (0.99) during the classification of ''Happy'' and ''Sad'' sentiments. It indicates that our method correctly identified false positive and false negative instances out of all given observations. Further, it leads to computing a high F1-score which showed a balanced tradeoff between positive and negative classes, providing a more comprehensive assessment of its effectiveness in imbalanced classification tasks. These measures can be easily computed using the confusion matrix shown in Figure 6. In this case, the ordering of the methods is LSTM+RF > NB > VLSTM ≈ SVM > GBM > ABA.

TABLE 6. Comparison of the proposed classification model (last row and last column) with different baseline methods in terms of various performance measures.

Comparison of the proposed classification model (last row and last column) with baseline methods in terms of Classification accuracy						
Sr. No	NB	GBM	SVM	VLSTM	ABA	LSTM+ RF
DUTSD	87.3%	91.34%	90.04%	93.18%	84.60%	-
NTUSD	80.92%	87.18%	92.50%	91.30%	78.54%	-
Deep LSTM Features	81.80%	83.04%	89.60%	94.80%	63.34%	99.19%
Comparison of the proposed classification model (last row and last column) with baseline methods in terms of Precision score						
Sr. No	NB	GBM	SVM	VLSTM	ABA	LSTM+ RF
DUTSD	0.59	0.62	0.58	0.73	0.69	-
NTUSD	0.64	0.86	0.52	0.64	0.55	-
Deep LSTM Features	0.72	0.66	0.68	0.78	0.58	0.99
Comparison of proposed classification model (last row and last column) with baseline methods in terms of Recall score						
Sr. No	NB	GBM	SVM	VLSTM	ABA	LSTM+ RF
DUTSD	0.42	0.58	0.64	0.60	0.54	-
NTUSD	0.58	0.74	0.52	0.62	0.62	-
Deep LSTM Features	0.82	0.68	0.72	0.62	0.66	0.99
Comparison of proposed classification model (last row and last column) with baseline methods in terms of F1-score						
Sr. No	NB	GBM	SVM	VLSTM	ABA	LSTM+ RF
DUTSD	0.49	0.60	0.608	0.658	0.60	-
NTUSD	0.60	0.7955	0.52	0.629	0.58	-
Deep LSTM Features	0.77	0.67	0.699	0.699	0.62	0.99
Comparison of proposed classification model (last row and last column) with baseline methods in terms of Kappa-score						
Sr. No	NB	GBM	SVM	VLSTM	ABA	LSTM+ RF
DUTSD	-0.38	0.52	0.64	0.78	0.24	-
NTUSD	0.68	-0.48	0.38	0.44	0.44	-
Deep LSTM Features	0.38	-0.14	0.50	-0.08	0.14	0.82
Comparison of proposed classification model (last row and last column) with baseline methods in terms of training Execution time (in seconds)						
Sr. No	NB	GBM	SVM	VLSTM	ABA	LSTM+ RF
DUTSD	34.22	20.56	32.40	72.14	22.80	-
NTUSD	44.12	22.94	18.80	48.60	36.30	-
Deep LSTM Features	38.92	28.52	16.34	51.60	36.2	102

In a different comparison, four methods (LSTM+ RF, NB, SVM, ABA) out of six approaches realized positive cappa scores while the remaining two computed negative values. Here, a high cappa score (0.82) of the proposed classification model indicated a high agreement between predicted and true class labels. In the last comparison, the proposed approach showed a very high execution time (102 seconds) which is also a main limitation of our methodology. In this case, SVM won the race and computed steady results in 16 seconds. The primary reason for this limitation is the deep structure of the

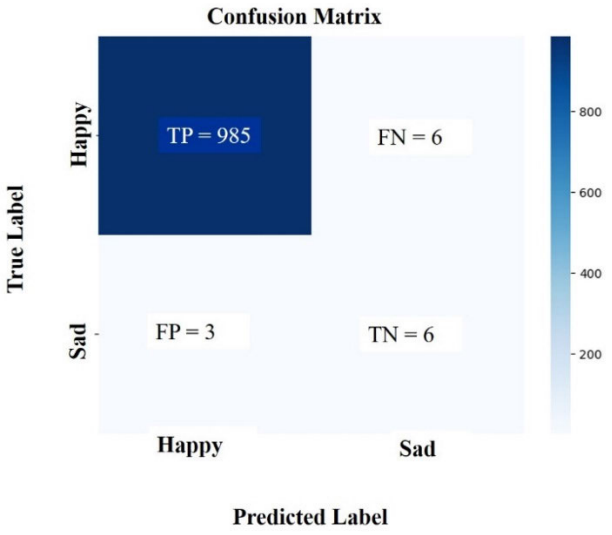


FIGURE 6. Confusion matrix for proposed classification framework.

LSTM model (8 hidden layers) used for feature extraction and the time-consuming hyperparameter tuning procedure in the RF classifiers. In the future, this limitation can be resolved by amalgamating new approaches such as batch processing [55], transfer learning [56], GPU acceleration [57], and. model compression [58]. The first approach, the batch processing concept in NLP classification tasks is aimed to improve the work rate by handling multiple cases at a go to reduce the time taken to serve the customer. Queries are submitted in batches to speed up data retrieval, documents are read and processed in a single step which employs parallel processing, and so most computation demands are reduced. Trimmed algorithms and focused batch operations allow for performance optimization while memory management remains in its place.

In the second optimization, the use of transfer learning for NLP classification speeds up the learning process by utilizing pre-trained models and knowledge thereby using less time for training from the beginning. Through fine-tuning data from special datasets, the process becomes more effective, so that the number of iterations that are needed and the amount of data used decrease. Also, transfer learning enables feature extraction from pre-trained models that in turn cuts data preprocessing costs. This method is especially profitable when dealing with scarce annotated data or when the model is to be adjusted to a new domain. GPU acceleration is one of the methods employed in fast NLP classification whereby the parallel processing capabilities of GPUs are harnessed to speed up the tasks. Such processors are the best options for executing many calculations simultaneously, which is why they are highly suitable for the matrix operations in deep learning neural networking models used for NLP.

During training, GPUs help yield faster iterations and iterations with various architectures and hyperparameter attributes producing quicker development of models. Furthermore, GPU-fastened libraries and frameworks significantly speed up the development and deployment of NLP models and boost their performance dramatically. GPUs can support model parallelism and distributed training which bring about more scalability and efficiency in scenarios dealing with large-scale NLP classification tasks. In our last suggested improvement, the Model compression approach allows for decreasing both inference time and memory use while making the sentiment analysis model work faster and easier to deploy on devices with limited resources. For example, pruning, quantization, and knowledge distillation are some of the methods on how to compress models without significant reduction in performance. Therefore, they are much more efficient for sentiment analysis applications that are time-based. It should be noted that the total execution time (102 seconds) of our approach includes both, the training and testing phase. So once trained, it can be used as an effective alternative to other competitive models.

Another consideration for our study's limitations is the small validation scope of our proposed methodology. This is due to our strategy being evaluated on a single dataset, which limits the generalizability of our findings. In the future, we will extend our study to broaden the validation scope of our methodology by including other datasets representing various scenarios or populations relevant to our study. This would improve the robustness and generalizability of our findings, allowing for a more comprehensive evaluation of the usefulness of our suggested approach in other circumstances. Furthermore, using techniques like cross-validation or external validation on independent datasets may improve the validity and dependability of our methodology. The emotional content and semantic structure of the stories may be slanted toward specific ethnic or demographic perspectives, depending on factors such as the authors' origins or the selection criteria for the stories. This bias may limit the model's capacity to accurately generalize to larger populations or cultural situations. Our models developed for children's books may not be applied to other genres of text, such as adult literature, social media posts, or news pieces. Each domain has unique language qualities and emotional nuances that necessitate model adjustments or retraining.

Although the dataset of 4000 children's stories can be used as a good source of emotional analysis, some steps may affect the model's capabilities. The focus of the dataset on the children's genre may be subjected to bias and increase the dependence on only one particular emotional expression and the linguistic style which can limit the model's generalizability to other domains and age groups. Additionally, most of the semantics in kid's narratives is based on emotions which again is a challenge for sentiment analysis algorithms to try to capture. Combining these challenges with the fact that sentiment analysis models are not very accurate when it comes to capturing subtleties and figures of speech makes it much more complicated. To tackle the challenges discussed, dataset diversity and model adaptation need to be given special thought. Preprocessing is another significant consideration. All these things contribute to the

implementation of the sentiment analysis framework and its effectiveness.

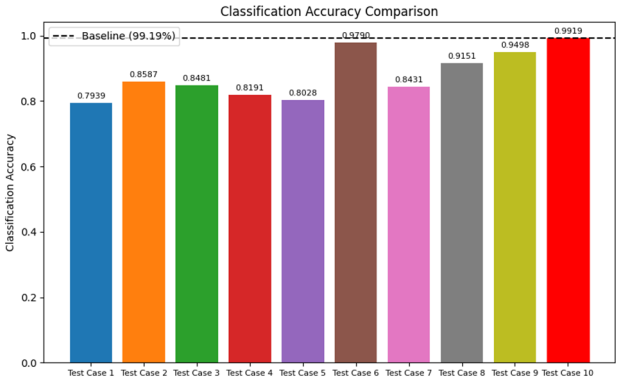


FIGURE 7. Classification accuracy levels of all test cases including the proposed one (in red).

In our experiment, we performed a comprehensive sensitive analysis in terms of various test cases adopted from different combinations of hyperparameters listed in Table 2. To explain the sensitivity performance of the proposed approach, the details of 10 test cases are discussed below, and respective classification accuracy scores are shown in Figure 7.

[1] **Test Case 1**: This configuration stands for a small LSTM network with 50 units per layer, a minute learning rate of 0. 0001, 0.5 dropout rates, 16 size of batch, Adam optimizer, ReLU activation function, categorical cross entropy loss function and early stopping turned on. The model effectiveness can be affected by the small LSTM size and low learning rate, which may take a longer time to converge and also show lower accuracy. Accuracy level: 85. 12%.

[2] **Test Case 2**: Similar to Test Case 1 but with a slightly higher learning rate of 0.01. 001. Thus, we may achieve a faster convergence as compared to Test Case 1, but it would be at the expense of the risk of overshooting the optimal solution. Accuracy level: 88. 76%

[3] **Test Case 3:** This architecture will have higher dropout rates and batch size than Test Case 1 which will likely help in addressing overfitting and improving generalization. Nevertheless, the efficiency of these hyperparameters would depend on the unique characteristics of the dataset. Accuracy level: 91. 34%

[4] **Test Case 4:** Similar to Test Case 3 but with a larger LSTM model size of 500 units per layer. This increased model capacity can capture intricate patterns in the data, thereby facilitating a higher level of accuracy. Accuracy level: 93. 21%

[5] **Test Case 5:** The architecture is composed of a different optimizer (Adam) and loss function (Mean Squared Error) from Test Case 1. The optimizer and loss function choice can considerably affect training dynamics and model performance. Accuracy level: 90. 55%

[6] **Test Case 6:** Like Test Case 1 but without early stopping. Therefore, this may take a long time until the machines overfit if it is not controlled well. Accuracy level: 84. 67%

[7] **Test Case 7:** In this setup, we use ReLU instead of the identity activation function, which contrasts Test Case 1. The choice of the activation function can be decisive for the model if it can or not capture non-linear relationships within the data. Accuracy level: 89. 98%

[8] **Test Case 8:** It will be similar to the Test Case and Batch Size of 128. This increasing batch size could allow the algorithm to learn more from the data and create more uniform gradients to obtain a faster convergence, though it could also lead to less diverse updates. Accuracy level: 92. 05%

[9] **Test Case 9:** This configuration is comprised of a different loss function (Mean Squared Error) just like in Test Case 1. The choice between loss functions can affect model behavior during the training process and help it optimize for accuracy. Accuracy level: 91. 78%

[10] **Test Case 10.** This case refers to the best configuration among all the test cases. Our proposed approach used this case to realize maximum classification accuracy score. It involves a set of hyperparameters that are tuned well to the dataset and the task, allowing for better model optimization. Accuracy level: 99. 19%.

V. CONCLUSION

Sentiment classification is a significant approach for opinion mining that assesses the views, opinions, and emotions obtained from text, speech, and big data through different methods. The feature extraction from a large dataset seems tedious and could incorporate human biases, impacting the feature quality and, in turn, eventually affecting the classification task. The redundant and insignificant features may get extracted, resulting in higher misclassification. Hence, the study propounded an efficient sentimental analysis model using Deep LSTM feature extraction with An updated Tuned RF classifier. The extraction of significant features gained through many layers of LSTM to learn the prominent features aids the classification of precise outcomes from pen-ultimate model layers.

The Weight updated hyperparameter RF classifier learns complex multi-variate functions in every mode to select relevant paths, hence capable of classifying precise features of emotions. The performance assessment of the proposed model, with its internal model outcomes, expounds high accuracy (0.99), precision (0.99), Recall (0.99), and F1-Score (0.99) values. Further to this, comparative analysis of the proposed method with other classifiers also yields outperforming outcomes with 99.19 % accuracy, revealing the model's efficiency. Regarding continuous complexity and the expanding size of big data, it seems crucial to consider time and reliability, specifically in other critical systems that necessitate rapid response. The present study can be extended by incorporating various data sources such as product reviews, customer feedback, and social media posts for handling a wider range of language features and nuances

in future studies. Moreover, SA can also be integrated into various applications such as recommendation engines, personalized marketing, and chat-bots as future work. In addition, researchers may work to reduce the misclassification rate by exploring data augmentation and ensemble methods during evaluation in sentiment analysis. Data variant enriches the train set by having its multiple copies with some of them shifted to have more grammatical patterns while the other are shifted to have more emotional expression. The ensemble methods though, utilize the wide range of approaches by combining the prediction of many models trained on different subsets of the data or by using multiple algorithms. Such diversity of models enhances the accuracy level and the prediction stability. The strategies can be designed to be included in the sentiment analysis pipeline and as a result, the model's robustness and ability to generalize can be improved which leads to more reliable prediction during testing and some level of reduction in misclassification. The proposed technique can also be enhanced for handling languages other than English. Furthermore, enhanced techniques like multi-semantic feature extraction and fusion technology can be considered as future work to improve the efficiency of the proposed model.

REFERENCES

[1] K. Pasupa, P. Netisopakul, and R. Lertsuksakda, "Sentiment analysis of Thai children's stories," *Artif. Life Robot.*, vol. 21, pp. 357–364, Jun. 2016.

[2] P. J. Stone, "Thematic text analysis: New agendas for analyzing text content," in *Text Analysis for the Social sciences*, 2020, pp. 35–54.

[3] X. Wang, G. Xu, Z. Zhang, L. Jin, and X. Sun, "End-to-end aspect-based sentiment analysis with hierarchical multi-task learning," *Neurocomputing*, vol. 455, pp. 178–188, Sep. 2021.

[4] A. R. W. Sait and M. K. Ishak, "Deep learning with natural language processing enabled sentimental analysis on sarcasm classification," *Comput. Syst. Sci. Eng.*, vol. 44, no. 3, pp. 2553–2567, 2023.

[5] C. F. Tsai, "Bag-of-words representation in image annotation: A review," *Int. Scholarly Res. Notices*, vol. 2012, Sep. 2012, Art. no. 376804.

[6] H. Christian, M. P. Agus, and D. Suhartono, "Single document automatic text summarization using term frequency-inverse document frequency (TF-IDF)," *ComTech, Comput., Math. Eng. Appl.*, vol. 7, no. 4, p. 285, Dec. 2016.

[7] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, "From word embeddings to document distances," in *Proc. Int. Conf. Mach. Learn.*, Jun. 2015, pp. 957–966.

[8] A. Pauls and D. Klein, "Faster and smaller N-gram language models," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics, Hum. Lang. Technol.*, Jun. 2011, pp. 258–267.

[9] A. Onan, S. Korukoğlu, and H. Bulut, "A multiobjective weighted voting ensemble classifier based on differential evolution algorithm for text sentiment classification," *Expert Syst. Appl.*, vol. 62, pp. 1–16, Aug. 2016.

[10] M. E. Mowlaei, M. Saniee Abadeh, and H. Keshavarz, "Aspect-based sentiment analysis using adaptive aspect-based lexicons," *Exp. Syst. Appl.*, vol. 148, Jun. 2020, Art. no. 113234.

[11] K. E. N. Kumar and V. Uma, "Intelligent sentinet-based lexicon for context-aware sentiment analysis: Optimized neural network for sentiment classification on social media," *J. Supercomput.*, vol. 77, no. 11, pp. 12801–12825, Nov. 2021.

[12] R. K. Mishra, S. Urolagin, J. A. A. Jothi, A. S. Neogi, and N. Nawaz, "Deep learning-based sentiment analysis and topic modeling on tourism during COVID-19 pandemic," *Frontiers Comput. Sci.*, vol. 3, Nov. 2021, Art. no. 775368.

[13] S. R. Ahmad, A. A. Bakar, and M. R. Yaakub, "A review of feature selection techniques in sentiment analysis," *Intell. Data Anal.*, vol. 23, no. 1, pp. 159–189, 2019.

[14] L. Duong, "Natural language processing for resource-poor languages," Univ. Melbourne., Parkville, VIC, Australia, Tech. Rep., 2017.

[15] M. S. B. Razali, "Figurative language detection using deep learning and contextual features," Tech. Rep., 2023.

[16] D. Wang, C. Gong, and Q. Liu, "Improving neural language modeling via adversarial training," in *Proc. Int. Conf. Mach. Learn.*, May 2019, pp. 6555–6565.

[17] G. Ciaburro and B. Venkateswaran, *Neural Networks With R: Smart Models Using CNN, RNN, Deep Learning, and Artificial Intelligence Principles*. Birmingham, U.K.: Packt, 2017.

[18] Q. Wang, R.-Q. Peng, J.-Q. Wang, Z. Li, and H.-B. Qu, "NEWLSTM: An optimized long short-term memory language model for sequence prediction," *IEEE Access*, vol. 8, pp. 65395–65401, 2020.

[19] R. K. Behera, M. Jena, S. K. Rath, and S. Misra, "Co-LSTM: Convolutional LSTM model for sentiment analysis in social big data," *Inf. Process. Manage.*, vol. 58, no. 1, Jan. 2021, Art. no. 102435.

[20] A. V. Geetha, T. Mala, D. Priyanka, and E. Uma, "Multimodal emotion recognition with deep learning: Advancements, challenges, and future directions," *Inf. Fusion*, vol. 105, May 2024, Art. no. 102218.

[21] M. E. Wall, A. Rechtsteiner, and L. M. Rocha, "Singular value decomposition and principal component analysis," in *A Practical Approach to Microarray Data Analysis*. Boston, MA, USA: Springer, 2003, pp. 91–109.

[22] F. Noroozi, T. Sapiński, D. Kamińska, and G. Anbarjafari, "Vocal-based emotion recognition using random forests and decision tree," *Int. J. Speech Technol.*, vol. 20, no. 2, pp. 239–246, Jun. 2017.

[23] A. T. Azar, H. I. Elshazly, A. E. Hassanien, and A. M. Elkorany, "A random forest classifier for lymph diseases," *Comput. Methods Programs Biomed.*, vol. 113, no. 2, pp. 465–473, Feb. 2014.

[24] D. Anguita, L. Ghelardoni, A. Ghio, L. Oneto, and S. Ridella, "The 'K' in K-fold cross validation," in *Proc. ESANN*, vol. 102, Apr. 2012, pp. 441–446.

[25] M. Ahmad, S. Aftab, S. S. Muhammad, and S. Ahmad, "Machine learning techniques for sentiment analysis: A review," *Int. J. Multidiscip. Sci. Eng.*, vol. 8, no. 3, p. 27, 2017.

[26] C. Colón-Ruiz and I. Segura-Bedmar, "Comparing deep learning architectures for sentiment analysis on drug reviews," *J. Biomed. Informat.*, vol. 110, Oct. 2020, Art. no. 103539.

[27] R. Fernandez-Beltran and F. Pla, "Prior-based probabilistic latent semantic analysis for multimedia retrieval," *Multimedia Tools Appl.*, vol. 77, pp. 16771–16793, Feb. 2018.

[28] M. Nilashi, R. A. Abumalloh, S. Samad, M. Alrizq, S. Alyami, H. Abosaq, A. Alghamdi, and N. A. M. Akib, "Factors impacting customer purchase intention of smart home security systems: Social data analysis using machine learning techniques," *Technol. Soc.*, vol. 71, Nov. 2022, Art. no. 102118.

[29] M. Abbas, K. A. Memon, A. A. Jamali, S. Memon, and A. Ahmed, "Multinomial Naive Bayes classification model for sentiment analysis," *Int. J. Comput. Sci. Netw. Secur.*, vol. 19, no. 3, p. 62, 2019.

[30] L.-C. Chen, C.-M. Lee, and M.-Y. Chen, "Exploration of social media for sentiment analysis using deep learning," *Soft Comput.*, vol. 24, no. 11, pp. 8187–8197, Jun. 2020.

[31] M. Ghosh and G. Sanyal, "An ensemble approach to stabilize the features for multi-domain sentiment analysis using supervised machine learning," *J. Big Data*, vol. 5, no. 1, p. 44, 2018.

[32] G. Wang and S. Y. Shin, "An improved text classification method for sentiment classification," *J. Inf. Commun. Converg. Eng.*, vol. 17, no. 1, pp. 41–48, 2019.

[33] J. Li and M. Sun, "Scalable term selection for text categorization," in *Proc. Joint Conf. Empirical Methods Natural Lang. Process. Comput. Natural Lang. Learn. (EMNLP-CoNLL)*, Jun. 2007, pp. 774–782.

[34] Y. Cai, Q. Huang, Z. Lin, J. Xu, Z. Chen, and Q. Li, "Recurrent neural network with pooling operation and attention mechanism for sentiment analysis: A multi-task learning approach," *Knowl.-Based Syst.*, vol. 203, Sep. 2020, Art. no. 105856.

[35] 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," *Inf. Process. Manag.*, vol. 56, no. 3, pp. 637–653, 2019.

[36] J. V. Tembhurne and T. Diwan, "Sentiment analysis in textual, visual and multimodal inputs using recurrent neural networks," *Multimedia Tools Appl.*, vol. 80, no. 5, pp. 6871–6910, Feb. 2021.

[37] C. Gan, Q. Feng, and Z. Zhang, "Scalable multi-channel dilated CNN-BiLSTM model with attention mechanism for Chinese textual sentiment analysis," *Future Gener. Comput. Syst.*, vol. 118, pp. 297–309, Jul. 2021.
[38] S. Mboutayeb, A. Majda, and N. S. Nikolov, "Multilingual sentiment analysis: A deep learning approach," in *Proc. 2nd Int. Conf. Big Data, Model. Mach. Learn.*, 2021, pp. 1–7.
[39] J.-L. Wu and W.-Y. Chung, "Sentiment-based masked language modeling for improving sentence-level valence–arousal prediction," *Int. J. Speech Technol.*, vol. 52, no. 14, pp. 16353–16369, Nov. 2022.
[40] S. Soubraylu and R. Rajalakshmi, "Hybrid convolutional bidirectional recurrent neural network based sentiment analysis on movie reviews," *Comput. Intell.*, vol. 37, no. 2, pp. 735–757, May 2021.
[41] R. S. Perdana and A. Pinandito, "Combining likes-retweet analysis and Naive Bayes classifier within Twitter for sentiment analysis," *J. Telecommun., Electron. Comput. Eng.*, vol. 10, nos. 1–8, pp. 41–46, 2018.
[42] W. Budiharto and M. Meiliana, "Prediction and analysis of Indonesia presidential election from Twitter using sentiment analysis," *J. Big Data*, vol. 5, no. 1, pp. 1–10, Dec. 2018.
[43] J. Carney and C. Robertson, "4000 stories with sentiment analysis dataset," Brunel Univ. London, London, U.K., Tech. Rep., 2019, doi: 10.17633/rd.brunel.7712540.v1.
[44] J. Carney and C. Robertson. *4000 Stories With Sentiment Analysis Dataset*. [Online]. Available: https://brunel.figshare.com/articles/dataset/4000_stories_with_sentiment_analysis_dataset/7712540
[45] Y. Liu, Z. Liu, T. S. Chua, and M. Sun, "Topical word embeddings," in *Proc. AAAI Conf. Artif. Intell.*, vol. 29, no. 1, 2015, pp. 1–9.
[46] N. Dehak, R. Dehak, J. R. Glass, D. A. Reynolds, and P. Kenny, "Cosine similarity scoring without score normalization techniques," in *Proc. Odyssey*, vol. 15, Jun. 2010, pp. 1–5.
[47] E. Loper and S. Bird, "NLTK: The natural language toolkit," 2002, *arXiv:cs/0205028*.
[48] C. Hutto and E. Gilbert, "VADER: A parsimonious rule-based model for sentiment analysis of social media text," in *Proc. Int. AAAI Conf. Web Social Media*, vol. 8, no. 1, May 2014, pp. 216–225.
[49] S. Sehgal, J. Sharma, and N. Chaudhary, "Generating image captions based on deep learning and natural language processing," in *Proc. 8th Int. Conf. Rel., INFOCOM Technol. Optim. (Trends Future Directions) (ICRITO)*, Jun. 2020, pp. 165–169.
[50] R. Ge, S. M. Kakade, R. Kidambi, and P. Netrapalli, "The step decay schedule: A near optimal, geometrically decaying learning rate procedure for least squares," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–9.
[51] T. T. Joy, S. Rana, S. Gupta, and S. Venkatesh, "Hyperparameter tuning for big data using Bayesian optimisation," in *Proc. 23rd Int. Conf. Pattern Recognit. (ICPR)*, Dec. 2016, pp. 2574–2579.
[52] A. Parmar, R. Katariya, and V. Patel, "A review on random forest: An ensemble classifier," in *Proc. Int. Conf. Intell. Data Commun. Technol. Internet Things (ICICI)*. Cham, Switzerland: Springer, 2018, pp. 758–763.
[53] L. Xu, H. Lin, Y. Pan, H. Ren, and J. Chen, "Construction of emotional vocabulary ontology," *J. Inf.*, vol. 27, no. 2, p. 180, 2008.
[54] C. C. Chen, H. H. Huang, and H. H. Chen, "NTUSD-Fin: A market sentiment dictionary for financial social media data applications," in *Proc. 1st Financial Narrative Process. Workshop (FNP)*, May 2018, pp. 37–43.
[55] J. W. Fowler and L. Mönch, "A survey of scheduling with parallel batch (p-batch) processing," *Eur. J. Oper. Res.*, vol. 298, no. 1, pp. 1–24, Apr. 2022.
[56] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *J. Big Data*, vol. 3, pp. 1–40, Jun. 2016.
[57] A. Athanasopoulos, A. Dimou, V. Mezaris, and I. Kompatsiaris, "GPU acceleration for support vector machines," in *Proc. 12th Interfaces Workshop Image Anal. Multimedia Interact. Services (WIAMIS)*, Delft, The Netherlands, vol. 164, 2011, pp. 1–9.
[58] T. Choudhary, V. Mishra, A. Goswami, and J. Sarangapani, "A comprehensive survey on model compression and acceleration," *Artif. Intell. Rev.*, vol. 53, no. 7, pp. 5113–5155, Oct. 2020.



AZHAR AHMED BILAL was born in Kirkuk, Iraq, in 1982. She received the B.Sc. degree in software engineering techniques from Northern Technical University, Kirkuk, in 2004. She is currently pursuing the master's degree with the Department of Computer Engineering, Gazi University. Since 2016, she has been with the Computer Center Department, Kirkuk University.



O. AYHAN ERDEM received the B.Sc., M.Sc., and Ph.D. degrees from the Gazi University Institute of Science and Technology, Turkey. In 1990, he attended the English Language Education Program, Indiana University, USA. He finished the Technology of Computing Education at Purdue University. He is currently a Professor with the Department of Computer Engineering, Gazi University. He has published books and numerous articles in international journals. His research interests include computer networks, programming languages, IT, computer systems, software engineering, and fuzzy logic.



SINAN TOKLU was born in Adana, Turkey, in 1979. He received the B.Sc. degree in computer engineering from Eastern Mediterranean University, in 2004, and the M.Sc. degree in computer engineering and the Ph.D. degree in electrical electronics education from Gazi University, Ankara, Turkey, in 2007 and 2013, respectively. From 2007 to 2010, he was a Research Assistant with Gazi University. From 2010 to 2014, he was a Software Developer with TEİAŞ, Turkey. From 2014 to 2021, he was an Assistant Professor with the Department of Computer Engineering, Düzce University. Since 2022, he has been an Associate Professor with the Department of Computer Engineering, Gazi University. His research interests include the Internet of Things, WSN, deep learning, software engineering, and smart city.