International Journal of Engineering and Technology Innovation, vol. 13, no. 3, 2023, pp. 251-264

# A Bi-Directional GRU Architecture for the Self-Attention Mechanism: An Adaptable, Multi-Layered Approach with Blend of Word Embedding

Amit Pimpalkar[1,2,*], Jeberson Retna Raj[1]

[1]School of Computing, Sathyabama Institute of Science and Technology, Chennai, India

[2]Department of Computer Science and Engineering, Jhulelal Institute of Technology, Nagpur, India

Received 06 April 2023; received in revised form 13 April 2023; accepted 01 May 2023

DOI: https://doi.org/10.46604/ijeti.2023.11510

## Abstract

Sentiment analysis (SA) has become an essential component of natural language processing (NLP) with numerous practical applications to understanding "what other people think". Various techniques have been developed to tackle SA using deep learning (DL); however, current research lacks comprehensive strategies incorporating multiple-word embeddings. This study proposes a self-attention mechanism that leverages DL and involves the contextual integration of word embedding with a time-dispersed bidirectional gated recurrent unit (Bi-GRU). This work employs word embedding approaches GloVe, word2vec, and fastText to achieve better predictive capabilities. By integrating these techniques, the study aims to improve the classifier's capability to precisely analyze and categorize sentiments in textual data from the domain of movies. The investigation seeks to enhance the classifier's performance in NLP tasks by addressing the challenges of underfitting and overfitting in DL. To evaluate the model's effectiveness, an openly available IMDb dataset was utilized, achieving a remarkable testing accuracy of 99.70%.

**Keywords:** Bi-directional GRU, attention mechanism, deep learning, natural language processing, word embedding

## 1. Introduction

Assessing individual perceptions and empathy is fundamental to many aspects of human life. With the abundance of surveys and various hypotheses in cyberspace, it is imperative to computerize the most common method of extracting meaningful data. However, automatic sentiment analysis (SA) and sentiment classification (SC) pose a significant challenge for machines, as daily language is highly nuanced and difficult to process when inscribed in a semi-organized structure like XML. Opinion mining (OM) and text classification are among the fastest-growing areas of exploration in the fields of artificial intelligence (AI) and machine learning (ML) using natural language processing (NLP). The bag-of-words (BoW) model is widely used in text-handling applications to demonstrate texts mathematically.

Until recently, it was a common practice, for example, in GloVe, word2vec, and fastText, to inform representations stating that each word must capture all possible connotations. Mao et al. [1] proposed a technique for SC tasks that involves blending word vectors by reconciling logical words given by word2vec's words provided by the word reference. The trial demonstrated that using hybrid word vectors yielded significant improvements in accuracy. A straightforward technique is to consider each word as a one-hot vector with a length equal to the amount of the vocabulary and just one aspect set to 1, with all others set to 0. Nevertheless, one-hot word depiction captures the records of words in a lexicon, failing to reflect the dictionary's intricate social composition. Dangi et al. [2] introduced a chaotic coyote optimization algorithm (COA) based temporal weight-AdaBoost support vector machine (SVM) strategy for achieving exact classifications in context.

* Corresponding author. E-mail address: amit.pimpalkar@gmail.com

English language proofreader: Chih-Wen Teng

Term frequency-inverse document frequency (TF-IDF) is an ML technique that uses a statistical shift to determine the application of phrases in a text. The text may take the form of a document or a corpus. TF-IDF combines two benchmark metrics: TF and IDF [3]. People use the TF-IDF technique in straightforward ML and NLP scenarios because it is better suited for data retrieval, keyword ancestry, stop-word elimination, and fundamental text deconstruction.

However, TF-IDF cannot efficiently obtain the semantic content of phrases in a sequence. To address the limitations of TF-IDF, Mikolov et al. [4] developed Word2vec in 2013. This publicly available tool is rooted in deep learning (DL) and was specifically designed to overcome computational challenges and bottlenecks associated with DL models. The main characteristic is to arrive at the surrounding phrases using ML techniques like recurrent neural networks (RNN) or deep neural networks (DNN). It has two strains in which it trains miniatures using the continuous Bow (CBoW) or the skip-gram (SG) algorithms on the input texts. The weakness of word2vec falsehoods is that they have a style of original context window, which may forget global word-to-word confederations and high exposure tends to learn devilishly immoderate weights [5].

GloVe word embedding is an unsupervised method Pennington et al. and other Stanford researchers developed [6]. A count-based expression representation tool employs broad-overall statistics, which use co-occurrence data between words for expression representation. GloVe combines the benefits of two methods: expression vector learning matrices factorization strategies such as latent semantic analysis (LSA) and local context window methods such as SG. This matrix's construction blocks indicate how the expression *i* appears in the context of the expression *j*. In NLP, attention is increasingly becoming a common mechanism utilized in various neural network architectures. The encoder-decoder paradigm, which encodes the input sequence to a fixed-length vector, has faced some limitations.

To overcome the above-stated limitations, researchers proposed an attention vector (AV) that can dynamically evaluate the closeness of its connection to contextual principles and predict a building block that resembles an expression in a holding. The attention layer (AL) takes inspiration from human attention but is essentially a weighted mean reduction. The attention mechanism (AM) is the vital link connecting the encoder and decoder, providing the decoder with information from each encoder's hidden state. This structure allows the algorithm to effectively handle long input sentences by concentrating on essential portions of the input series and understanding their relationship.

Moreover, in a study by Dessí et al. [7], convolution neural network (CNN), using the word2vec model, performed better on news stories than tweets, as news pieces are usually more uniform. In recent years, the long short-term memory (LSTM) model has emerged as a prominent DL approach for organizing sequential information. When combined with an AM system, LSTM has demonstrated enhanced performance. Consequently, extensive investigations have been conducted to explore the fundamental architecture of LSTM [8-10], aiming to further improve its efficiency.

Furthermore, LSTM has also attained outstanding accomplishments in text classification. Researchers have tinkered with various word embedding techniques such as BoW, TF-IDF, and word2vec to scrutinize the performance of SA on the internet movie database (IMDb) dataset and observed that both TF-IDF and BoW outperformed word2vec, with TF-IDF edging out BoW [10]. The conventional CNN [8, 11] and RNN [8] solely employ the word-level function and are devoid of semantic components. These run-of-the-mill models possess an exceedingly limited capability, particularly for terms that depend on parameters to determine their significance. Hence, expanding the channels and utilizing a hierarchical configuration to process several contextual cues is imperative.

The manuscript is organized as follows: Section 2 presents the literature review. Section 3 describes the proposed method and architecture. Section 4 discusses datasets, experimental setups, and evaluation indices. Results and analysis for the proposed model are discussed in Section 5, and Section 6 concludes this research and outlines future work.

## 2. Literature Review

Amidst the plethora of studies on SA using ML, several researchers have proposed innovative techniques to overcome the limitations of traditional models. Meanwhile, the extensive meanings of each word in the text differ based on the contextual cues given. Certain words contribute more to the process, while others contribute less, and the sentiments conveyed by each word are predominantly focused [12]. Although Bidirectional-LSTM (BiLSTM) [13] can apprehend the contextual information of the text, it is implausible to pinpoint the pivotal data in the context-oriented data. Merging BiLSTM and AMs can further enhance the performance of SC. Li et al. [14] devised an adaptive cross-contextual word embedding method to learn custom embeddings for polysemic words in different contexts. Another researcher, Ltaifa et al. [15] introduced a Hybrid-Regularized-Autoencoder that uses flexible regularization to enhance feature selection without manual intervention.

On the other hand, Guo et al. [16] proposed a novel term coefficient approach to address issues with traditional models. This study combined TF-IDF weighted GloVe word embedding with CNN-LSTM architecture to develop a DL-based approach to SA on Twitter consumer reviews to demonstrate this approach's effectiveness. Another researcher, Jang et al. [17] used word2vec, CBoW, and SG word embedding techniques to distinguish between connected and random news stories and tweets. The CBoW model outperformed the SG model regarding accuracy and consistency, particularly in media articles. Meanwhile, Wang et al. [18] explored word embeddings involving CNN and BiLSTM encoders for SC. These novel approaches offer a promising solution to the challenges of SA using ML techniques.

Several researchers [19-21] employed LSTM, BiLSTM, and CNN models with pre-prepared GloVe embedding and learned that they were better than word2vec. Their model outperformed existing models due to boundary values and word embeddings that helped accomplish the best possible performance contrasted with different investigations. They analyzed word embedding and neural organization-based ways to deal with SC. A DNN based on a mix of CNN and BiLSTM was proposed and characterized for detection and classification framework based on anopheles search-optimized AlexNet convolution [22-23]. To tackle the Web service categorization challenge, Kang et al. [24] developed a topical attention-based BiLSTM model. They combined the local implicit state vector of BiLSTM with the global hierarchical Dirichlet process (HDP) topic vector.

Table 1 Comparison of the existing model performance on various datasets along with model accuracy

| Year | Ref. | Embedding method | Learning method | Dataset | Best performance |
|---|---|---|---|---|---|
| 2022 | [2] | GloVe, word2vec, fastText | TW-ADASVM-MCOA | Amazon food, Apple products | 87.45% |
| 2021 | [4] | GloVe | HAN, BiGRU, GNN | MEDLINE, OHSUMED | 67.50% |
| 2022 | [5] | GloVe | MBiLSTM | IMDb | 93.55% |
| 2019 | [11] | Word2vec | LSTM, BiLSTM, AC-BiLSTM | MR, IMDb, SST-1, SST-2 | 91.80% |
| 2019 | [17] | CBoW, SG | CNN | News articles for disease | 93.41% |
| 2020 | [18] | Word2vec, GloVe, fastText, ELMo, BERT | BiLSTM, CNN | SST-2, 20NewsGroup, AAPD, Reuters | 90.01% |
| 2021 | [19] | TF-IDF, GloVe | BiLSTM, CNN, ACL-SA | Sentiment140, US-Airline, SD4A | 94.53% |
| 2020 | [20] | Word2vec | BiLSTM, CRF, CNN | JNLPBA, NCBI-Disease | 86.93% |
| 2021 | [21] | SG, CBoW, GloVe | BiLSTM, ALSTM, SVM | SemEval | 88.75% |
| 2022 | [25] | GloVe, word2vec, fastText | CNN-LSTM, CoSE-T | SemEval, SST-2, SST-5, MR | 90.80% |
| 2022 | [26] | GloVe, word2vec, fastText | CNN | UCI KDD | 97.20% |
| 2022 | [27] | TextBlob + BoW & TF-IDF | SVM, CNN-LSTM, GRU | IMDb | 92.00% |
| 2021 | [28] | GloVe, word2vec, fastText | RNN, GL-RNN, UGRNN | Amazon.com | 93.75% |
| 2021 | [29] | Doc2Vec, BoW, TF-IDF | NB, SVM, DT, RF, k-NN | Amazon, Umich | 77.44% |

Furthermore, they resolved the class imbalance concerns using a modified COA, the chaotic COA. They evaluated the performance of their suggested approach using the SVM technique. Many researchers focused on word embeddings such as word2vec, GloVe, and fastText and contextualized embeddings such as embeddings from language models (ELMo) and bidirectional encoder representations from transformers (BERT). Through this methodology, the organization creates different morpheme-like vectors characterized by the actual linguistic language units of significance. Table 1 presents a comparative analysis of existing models' performance in terms of accuracy across multiple datasets.

All the above-studied strategies have made an extraordinary commitment to the most common method for learning the significance of words. However, because of the rapid rate of development in this field, a comprehensive review of attention is still lacking. This examination aims to assess different opinion investigation approaches using DL mechanisms, including CNN, LSTM, BiLSTM, and bidirectional gated recurrent unit (Bi-GRU) time distributed hierarchical attention network (TDHAN) models, and evaluate their exhibition on three-word embedding methods based on GloVe, word2vec, and fastText. The study is deeply rooted in the past, but research suggests a method for acquiring the significance of words from a different perspective. The paper gives the accompanying novelties and contributions:

(1) Three DL models that used dense CNN, LSTM, and BiLSTM to distinguish different degrees within internet-based text comments on movies were examined.

(2) The usage of three-word embedding methods that encode the opinions of messages in a persistent word representation based on GloVe, word2vec, and fastText was assessed.

(3) A DL Bi-GRU-TDHAN model is proposed to analyze word-level and sentence-level representations.

(4) An AM is integrated at the review text's word and semantic sentence levels.

(5) DL models are compared and contrasted with traditional baselines in text-based classification studies, and the results of the experiments reveal that the proposed model consistently outperforms other state-of-the-art methodologies.

(6) The advantages of consolidating word embedding representations with DL frameworks rather than standard ML models are shown. This research will help e-learning partners gain distinctive insights and shape future exploration in this area.

## 3. The Proposed Approach

The proposed model, Bi-GRU and transformer-based network TDHAN for SA is shown in Fig. 1 and constructed on the late-acquainted Transformer model, which has given substantial advancements for machine translation tasks. Contrasting RNN or CNN models, the transformer is qualified to learn dependencies amongst far-off places. Therefore, this research shows that attention-based models can perform different NLP responsibilities, such as obtaining allocated representations and SA, and are qualified to improve broad-brush closeness. The architecture and process of the Bi-GRU TDHAN model are encapsulated in the following steps.

Step 1: Several text pre-processing methods are used at the start of the process to clean up the text and break it into tokens.

Step 2: Use contextual expression representations as an embedding layer to define a text as real-valued vectors with GloVe, word2vec, and fastText.

Step 3: After embedding the text in real-valued vectors, the transformer network uses the self-attention process to arrange the input sequence into the hidden layer.

Step 4: The word encoder expression places Bi-GRU to learn factual representations of expressions to assess the interdependency between word representations.

Step 5: To obtain meaningful information from the sentence, word-level attention is used.

Step 6: The sentence-level encoder places a Bi-GRU to learn the considerable representation of the sentence, and attention is used to learn imperative sentences, among other sentences.

Step 7: Combination representations of the input are subsequently forwarded to a completely concatenated neural network.

Step 8: Eventually, a dense and Softmax layer is employed as an adjudicator of the text toward the SA classifier.

### *3.1. Word and sentence encoders*

The word and sentence encoders' layer aims to extract essential contexts from each word and statement. Consider that a document has $L$ sentences $s_i$, and each sentence contains $T_i$ words. A Bi-GRU is used as an encoding technique in this model to provide annotations for words by summing input from both sides, resulting in a summary variable $h_{it}$ for word attention whereas $h_i$ for sentence attention. A GRU model has a hidden state that serves as a memory unit for data transfer. Given the phrase vectors $s_i$, one can get a document vector, and $s_i$ summarizes the neighboring phrases around sentence $i$.

$$\vec{h}_{it} = \overrightarrow{GRU}\left(x_{it}\right),\ t \in [1,T] \tag{1}$$

$$\overleftarrow{h}_{it} = \overleftarrow{GRU}\left(x_{it}\right),\ t \in [T,1] \tag{2}$$

$$h_{it} = \left[\vec{h}_{it}, \overleftarrow{h}_{it}\right] \tag{3}$$

$$\vec{h}_{i} = \overrightarrow{GRU}\left(s_{i}\right),\ i \in [1,L] \tag{4}$$

$$\overleftarrow{h}_{i} = \overleftarrow{GRU}\left(s_{i}\right),\ i \in [L,1] \tag{5}$$

$$h_{i} = \left[\vec{h}_{i}, \overleftarrow{h}_{i}\right] \tag{6}$$

An annotation for a given word and sentence is acquired by concatenating the forward hidden state and backward hidden state, which outlines the information of the whole sentence as shown in Eqs. (3) and (6).

### *3.2. Word and sentence attention*

The word and sentence AL ensure that the network does not fail due to a tanh function. This function "corrects" input values within -1 and 1 and converts zeros to near-zero values. The idea is that the model should learn by training with randomized control network biases and weights and represent itself as $u_{it}$ for word attention whereas $u_i$ for sentence attention. During the training phase, the word contextual vector, $u_w$ is initialized randomly and jointly acquired. The sentence vector $s_i$ and $v$ is the sum of these actual weights combined with the previously derived context annotations for word and sentence attention, respectively.

The weighted sum of these hidden states is used to calculate the AV $u_{it}$ for AL 1, as follows:

$$u_{it} = \tanh\left(W_w h_{it} + b_w\right) \tag{7}$$

where the hidden state $h_{it}$ can be obtained at timestamp *t*, and the weighted parameter $\alpha_{it}$ is calculated using the following:

$$a_{it} = \frac{\exp\left(u_{it}^T u_w\right)}{\sum_t \exp\left(u_{it}^T u_w\right)} \tag{8}$$

The $s_i$ AV for AL *i* is calculated as follows; subsequently, all layers of attention are combined.

$$s_i = \sum a_{it} h_{it} \tag{9}$$

$$s = Concatnate\left(s_1 s_2\right) \tag{10}$$

Similarly, computation is done for word contextual vector $u_i$ and $v$ as follows:

$$u_i = \tanh\left(W_s h_i + b_s\right) \tag{11}$$

$$a_i = \frac{\exp\left(u_i^T u_s\right)}{\sum_i \exp\left(u_i^T u_s\right)} \tag{12}$$

$$v = \sum_i a_i h_i \tag{13}$$

Classification and the loss function for the model are described below,

$$p = soft\max\left(W_{c^v} + b_c\right) \tag{14}$$

$$L = -\sum_d \log\left(p_{dj}\right) \tag{15}$$

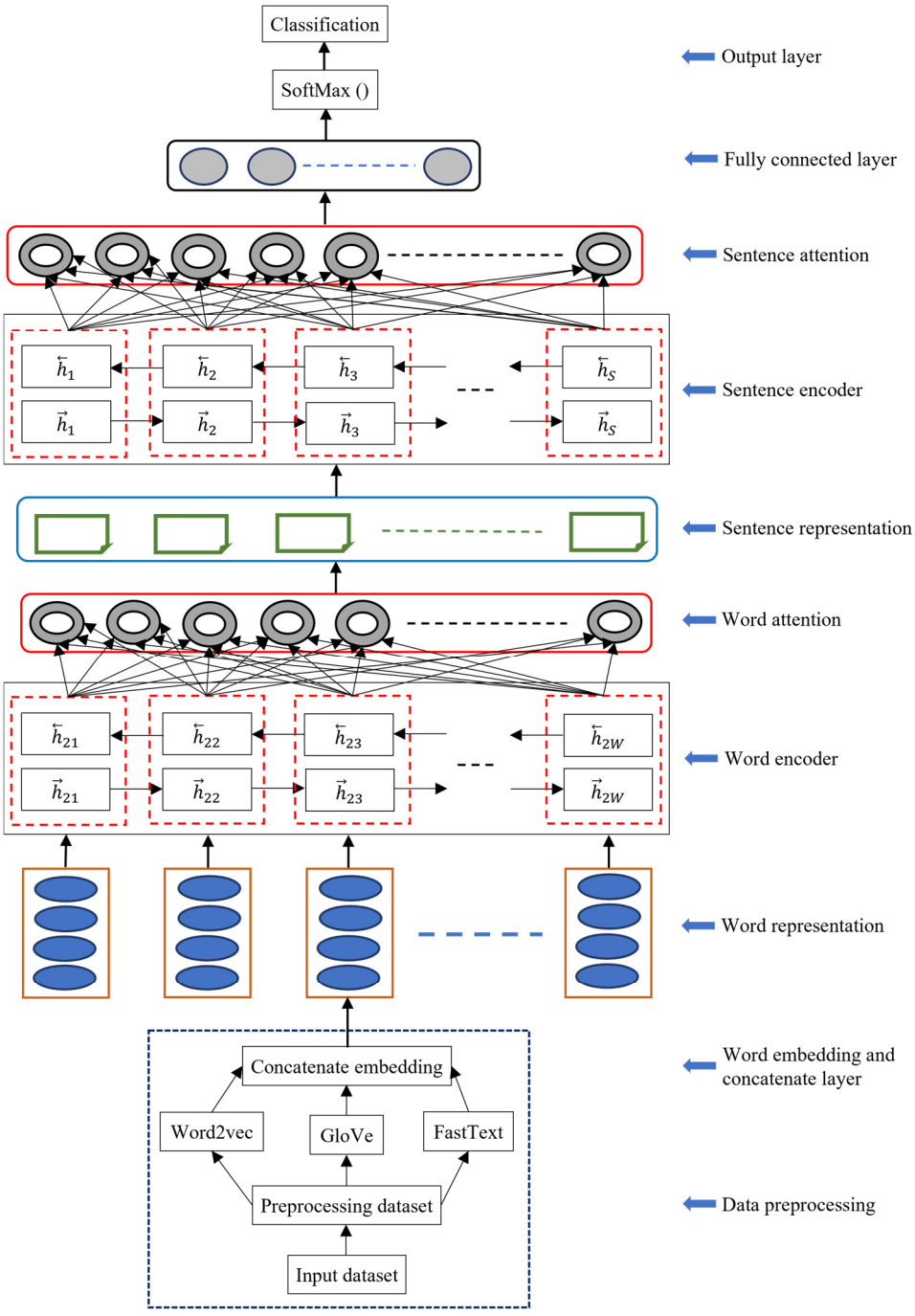



Fig. 1 Bi-GRU-TDHAN model architecture

### 3.3. Self-Attention layer

The Self-AL computes attention loads between each brace of tokens in an unattached single sequence and can capture long-range dependence further directly than its RNN co-equal. This allows each location in the encoder to headlong permeate all locations in the preceding layer and all locations in the input sequence in the earliest layer. The purpose of this was to average the characteristics of several components. In other words, the network needs to determine which inputs to "attend" to more than others on a dynamic basis.

The self-attention matrix-given input matrices (Q, K, V) are derived mathematically as follows:

$$Attention(Q,K,V) = soft\max\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{16}$$

where $QK^r/\sqrt{d_k}$ is a dot-product similarity scoring function.

The Softmax function can sometimes be hypersensitive to extremely high input values, which may strangle the gradient, slowing or completely stopping learning. Due to the average value of the dot product rising with the embedding dimension $K$, it is helpful to throttle it back somewhat to prevent the Softmax inputs from becoming too excessive. The transformer uses the "multi-head" attention mode to boost the influence of the attention unit and strengthen the model's computational capability for diverse situations. In "encoder-decoder attention" layers, the queries come from the preceding decoder layer, and the memory terms and values come from the output of the encoder. This enables every location in the encoder to watch over all points in the input sequence.

Instead of conducting a self-attention function with d-dimensional keys, values, and queries, it is advantageous to linearly project the queries, keys, and values h times with distinct linear projections to $dq$, $dk$, and $dv$ dimensions, respectively. Multi-head attention enables the model to simultaneously listen to input from various representations at varying locations with a solitary concentration on averaging attention results. The key, value, and query vectors are divided into multiple heads and projected; then, the separate layers go through a self-AM. Afterward, all the parts are merged into a single representation, as shown in Figs. 2 and 3; the calculation of the multi-headed self-AL is explained in the following steps:

$$Multi\ Head\ Attention(Q,K,V) = Concat(head_1, head_2, \ldots, head_n)W_o \tag{17}$$

where $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$ and $W_o$ represent a projection matrix for the multi-head output.

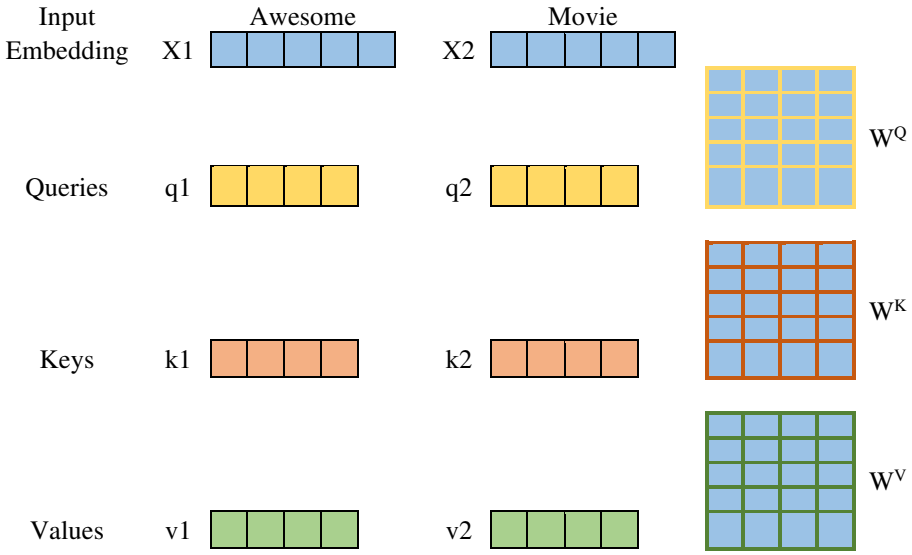



Fig. 2 Query, key, and value weights used to parameterize the self-AM

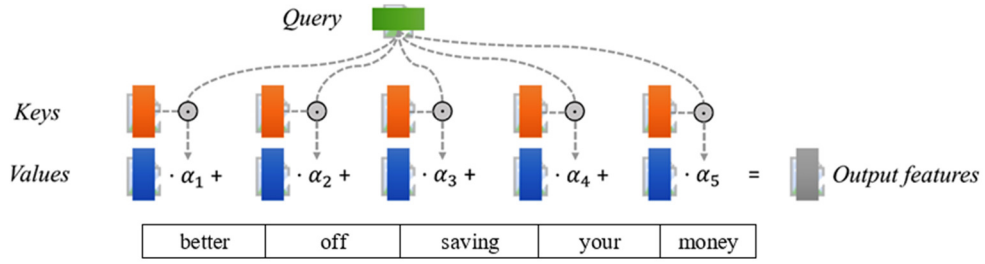



Fig. 3 Illustration showing the focus of attention on a phrase or sentence

### *3.4. Bi-GRU layer*

The Bi-GRU layer is a key component in proposed models, which consists of two GRU layers that process the input sequence in both the forward and backward directions. This dual-directional processing allows the model to capture information from both past and future time steps, facilitating a comprehensive understanding of the input sequence. To collect text context-dependent information, bidirectional unit gates are used.

Since a given distance may separate the adequate text information, a general filter may only notice local information and cannot comprehend the overall design features. The Bi-GRU gating network has a specific memory capacity due to its unique gate topology, which is ideal for lengthy text modeling. By incorporating information from both directions, the Bi-GRU layer enables the model to effectively analyze sequential patterns and capture long-term dependencies within the data. The outputs of the forward and backward GRU layers are typically combined through concatenation to create a representation that incorporates information from both directions. This combined representation is then passed on to subsequent layers for further processing. The Bi-GRU layer allows the model to capture the temporal structure of movie reviews, considering the influence of preceding and subsequent words on the sentiment expressed in a given sentence. This capability enhances the model's ability to predict sentiment accurately by leveraging a more distinct understanding of the sequential nature of the text.

## 4. Experimental Setup

In this segment, the dataset's detailed statistics and embedding methodology are briefly discussed for experimentation, followed by the pre-processing approaches before presenting the dataset to DL models. The experimental setup and assessment indices adopted for the study are represented schematically.

### *4.1. Datasets and pre-processing*

In this work, the results are compared considering the IMDb benchmark movie reviews dataset comprising the 50,000 tagged data as positive and negative in equal proportion. The dataset was initially split into training, validation, and testing sets as 8:1:1. Below, the datasets are described in more detail in Tables 2 and 3.

Table 2 Detailed statistics of the dataset

| Dataset | Max length | Min length | Avg length | Positive | Negative | Total |
|---|---|---|---|---|---|---|
| IMDb | 13,704 | 6 | 1,308 | 25,000 | 25,000 | 50,000 |

Table 3 First five reviews and corresponding label variables of the dataset without pre-processing

| Id | Label | Review |
|---|---|---|
| 1 | 1 | One of the other reviewers has mentioned that... |
| 2 | 1 | A wonderful little production. <br /><br />The... |
| 3 | 1 | I thought this was a wonderful way to spend ti... |
| 4 | 0 | Basically, there's a family where a little boy... |
| 5 | 1 | Petter Mattei's "Love in the Time of Money" is... |

The data was pre-processed with the help of a regular expression module, an underlying Python component. The following information activities were used: reducing text to lower case, removing Twitter handles @mention, removing non-ASCII characters such as accentuation, unique descriptions, and numbers, removing whitespace, hashtags, short and stop words, and tokenization [3, 5].

### 4.2. Experimental hyper-parameter

The current work trains a model with input layers on top of the embedding layer acquired via GloVe, word2vec, and fastText. The dimensions and real word vectors of these embeddings are explored, and the results are presented in Table 4. The model hyper-parameters are shown in Table 5. Python 3.7, TensorFlow 1.14, and Keras 2.3 libraries were used in this work and introduced on Windows 10. State-of-the-art DL models were trained on a Kaggle kernel with an Nvidia P100 GPU card, GPU memory of 16 GB, 12 GB of RAM, 1.32 GHz memory clock, and performance of 9.3 TFLOPS.

Table 4 Statistics of the embedding methods

| Embedding | Dimension | Total word vectors |
|---|---|---|
| GloVe | 200 | 11,93,514 |
| Word2vec | 300 | 30,00,000 |
| FastText | 300 | 9,99,995 |

Table 5 Hyper-parameter values used for the experimentation

| Hyper-parameter | Designated values |
|---|---|
| Max review length | 100 |
| Max sentence length | 15 |
| Max number of words | 20000 |
| Batch size | 100, 200, 512 |
| Number of epochs | 10 |
| Recurrent dropout | 0.3 |
| Optimizer | Adam |
| Activation function | Softmax |
| Batch normalizations | Yes |
| Loss function | Binary cross entropy |
| Output function | Sigmoid |

### 4.3. Evaluation index

The analysis of phrase structure uses accuracy, recall, and $F_1$ score value for evaluation:

(1) Accuracy: The accuracy rate in phrase structure analysis refers to the percentage of the number of correct phrases in the analysis result to the number of phrases in the analysis result. It promptly notifies us whether the model is being trained appropriately and how it will perform.

$$Accuracy = \frac{True\ positive + True\ negative}{True\ positive + True\ negative + False\ positive + False\ negative} \tag{18}$$

(2) Sensitivity: The sensitivity rate in phrase structure analysis refers to the percentage of the number of correct phrases in the analysis result to the total number of phrases in the test set.

$$Sensitivity = \frac{True\ positive}{True\ positive + False\ negative} \tag{19}$$

(3) Precision: Besides measuring model performance, precision can be defined as the likelihood that a randomly picked item labeled “relevant” is a true positive.

$$Precision = \frac{True\ positive}{True\ positive + False\ positive} \tag{20}$$

(4) $F_1$ score: Both precision and sensitivity must be adequate for a model to be termed “excellent.” $F_1$ score combines precision and sensitivity into a single number using the following formula.

$$F_1\ score = 2 \times \frac{Precision \times Sensitivity}{Precision + Sensitivity} \tag{21}$$

Confusion matrixes or error matrixes are two-dimensional matrices that depict the algorithm's performance. Comparing the number of accurate and erroneous forecasts made by the model against the fundamental categories in the test set shows the type of mistakes made by the model. Four possible implications could arise while conducting classification predictions:

(1) True positive: Number of occurrences that are indeed positive and estimated to be positive.

(2) True negative: Number of events that are truly negative and are anticipated negatively.

(3) False positive: Number of events that are truly negative but projected positive. These mistakes are also termed type 1 errors.

(4) False negative: Number of events that are truly positive but estimated negative. These mistakes are also termed type 2 errors.

Positives and negatives pertain to the forecast itself. True and false corresponds to the correctness of the predictions.

## 5. Experimental Analysis and Discussion

Five experiments were conducted to determine the best architecture for sentiment analysis, comparing different models. Unidirectional models like LSTM and CNN, as well as bidirectional models like BiLSTM and Bi-GRU were tested. The effectiveness of the TDHAN model was thoroughly examined, and the results are presented in Experiments 1-5, providing a comprehensive description of the findings. The proposed approach was evaluated, and its potency was compared to the leading-edge approaches using the analysis of movie reviews from IMDb, as depicted in Table 1.

(1) Experiment-1: Unidirectional model—LSTM

Table 6 Performances of the experimental dataset with unidirectional LSTM model

| Model | Test accuracy | Val accuracy | Variance in accuracy % |
|---|---|---|---|
| LSTM without embedding | 85.07 | 84.54 | 0.62 (Fit) |
| LSTM + GloVe | 94.64 | 88.86 | 6.11 (Fit) |
| LSTM + Word2vec | 98.30 | 86.71 | 11.79 (Overfit) |
| LSTM + FastText | 99.50 | 84.11 | 15.47 (Overfit) |
| LSTM + GloVe + FastText | 99.66 | 84.57 | 15.14 (Overfit) |
| LSTM + GloVe + FastText + Word2vec | **99.71** | 85.00 | 14.75 (Overfit) |

(2) Experiment-2: Unidirectional model—CNN

Table 7 Performances of the experimental dataset with unidirectional CNN model

| Model | Test accuracy | Val accuracy | Variance in accuracy % |
|---|---|---|---|
| CNN without embedding | 78.11 | 82.83 | -6.04 (Underfit) |
| CNN + GloVe | 90.36 | 88.53 | 2.03 (Fit) |
| CNN + Word2vec | 97.85 | 87.89 | 10.18 (Overfit) |
| CNN + FastText | 99.87 | 85.40 | 14.49 (Overfit) |
| CNN + GloVe + FastText | 99.45 | 84.86 | 14.67 (Overfit) |
| CNN + GloVe + FastText + Word2vec | **99.81** | 85.69 | 14.15 (Overfit) |

(3) Experiment-3: Bidirectional model—BiLSTM

Table 8 Performances of the experimental dataset with the BiLSTM model

| Model | Test accuracy | Val accuracy | Variance in accuracy % |
|---|---|---|---|
| BiLSTM without embedding | 87.36 | 86.49 | 1.00 (Fit) |
| BiLSTM + GloVe | 94.99 | 88.54 | 6.79 (Fit) |
| BiLSTM + Word2vec | 85.67 | 88.06 | -2.79 (Underfit) |
| BiLSTM + FastText | 99.49 | 85.17 | 14.39 (Overfit) |
| BiLSTM + GloVe + FastText | 99.49 | 84.66 | 14.91 (Overfit) |
| BiLSTM + GloVe + FastText + Word2vec | **99.62** | 85.57 | 14.10 (Overfit) |

(4) Experiment-4: Bidirectional GRU model—TDHAN

Table 9 Performances of the experimental dataset with the Bi-GRU TDHAN model

| Model | Test accuracy | Val accuracy | Variance in accuracy % |
|---|---|---|---|
| TDHAN + Bi-GRU without embedding | 98.54 | 88.93 | 9.75 (Fit) |
| TDHAN + Bi-GRU + GloVe | 96.80 | 90.96 | 6.03 (Fit) |
| TDHAN + Bi-GRU + Word2vec | 97.49 | 89.58 | 8.11 (Fit) |
| TDHAN + Bi-GRU + FastText | 96.86 | 89.69 | 7.40 (Fit) |
| TDHAN + Bi-GRU + GloVe + FastText | 98.18 | 89.56 | 8.78 (Fit) |
| TDHAN + Bi-GRU + GloVe + FastText + Word2vec | **99.70** | 89.93 | 9.80 (Fit) |

(5) Experiment-5: Bidirectional model—TDHAN

Table 10 Performances of the experimental dataset with the TDHAN model

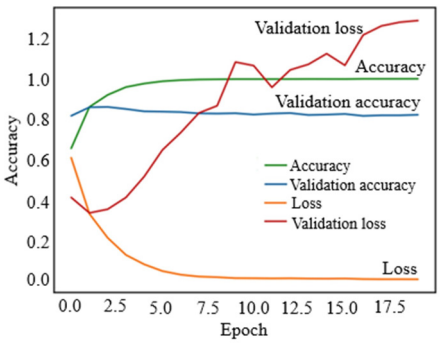
| Model | Test accuracy | Val accuracy | Variance in accuracy % |
|---|---|---|---|
| TDHAN without embedding | 99.16 | 88.04 | 11.24 (Overfit) |
| TDHAN + GloVe | 94.65 | 90.79 | 4.08 (Fit) |
| TDHAN + Word2vec | 95.75 | 89.86 | 6.15 (Fit) |
| TDHAN + FastText | 95.63 | 89.57 | 6.34 (Fit) |
| TDHAN + GloVe + FastText | 97.22 | 90.91 | 6.49 (Fit) |
| TDHAN + GloVe + FastText + Word2vec | **99.41** | 91.06 | 8.41 (Fit) |

To avoid overfitting and correctly evaluating the model, and as already discussed in Section 4.1 about the dataset, the train and validation test splits were used as 8:1:1. The cross-validation accuracy, testing accuracy, and bias difference between the validation and testing error were calculated. The variance between the testing and training errors was observed by running the model with different embedding variants of the system's GloVe, word2vec, and fastText under consideration. For this challenge, the optimum accuracy was believed to be 100% (human accuracy and ± 10% for underfit and overfit).

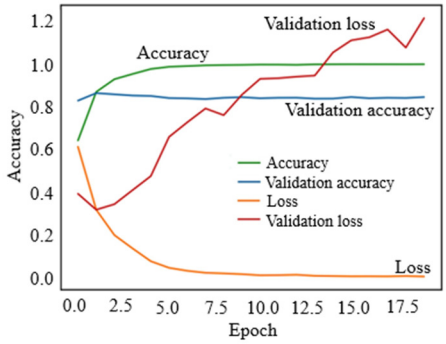
As can be seen from Experiments 1-3, as in Tables 6-8 for the models like LSTM, BiLSTM, and CNN, the power of embedding techniques improves the performance of the models almost every time, with a few exceptions. When comparing the accuracy variance, observations conclude that the models are either underfit or overfit, as shown in Figs. 4(a)-(c). This experimental analysis showed that the models are less flexible and cannot account for the data.

Consequently, models may fail to fit more data, lowering their ability to predict future observations. It has to be noted that even though the best performance of 99.81% was achieved with the CNN model with a blend of GloVe, fastText, and W2V embedding, as shown in Table 7, but lacks the overfitting issues over 14%, which make it more unstable for evaluation. On the other hand, in Experiments 4 and 5, as results shown in Tables 9 and 10, the performance of the proposed model not only improves over the previous experimental model, but the models are also much less prone to the underfitting and overfitting issues, and this is depicted in Figs. 4(d)-(e). The proposed models' confusion matrixes are shown in Fig. 5. This shows that the proposed model overcomes the issue mentioned earlier and becomes more stable regarding its ability to predict future observations.

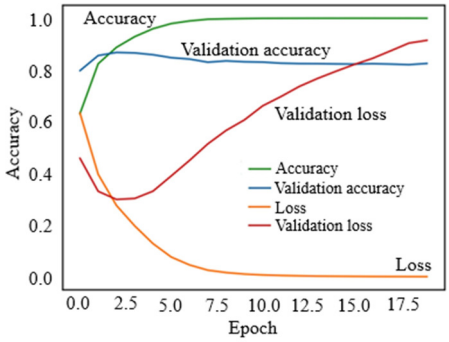
Nowadays, state-of-the-art embeddings such as BERT, ELMo, and XLNET have significantly achieved outstanding performance. In line with this, the performance of the proposed approach was compared against state-of-the-art embedding like XLNET [30]. The average accuracy rate of XLNET was 97.30%, while the experiment outperformed with 99.70%. To simplify the explanation and the complexity evaluation of the system, a few additional hierarchy-related notations are specified. In addition to sentence length $L$, the number of hierarchical levels $M$ and embedding sized $d$ as in Eq. (16) and $q$ indicate a unit of Softmax cost. Keeping in mind that $L$ and $d$ are typically data-dependent hyper-parameters and the execution time cost per iteration is $O$(d), the broader computational complexity of the TDHAN technique is $O$(dL).



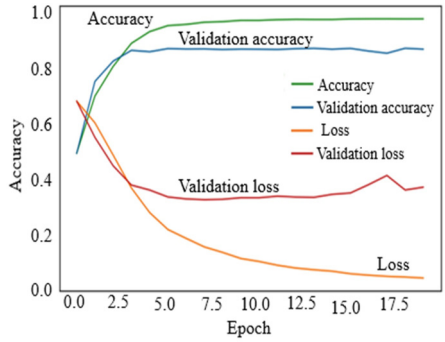

(a) LSTM model



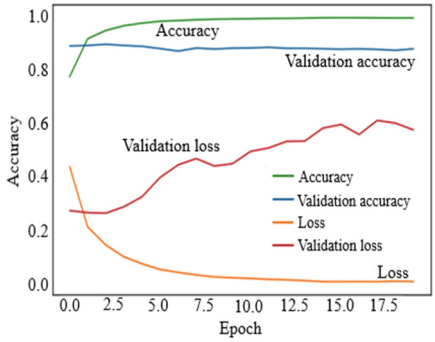

(b) BiLSTM model




(c) CNN model

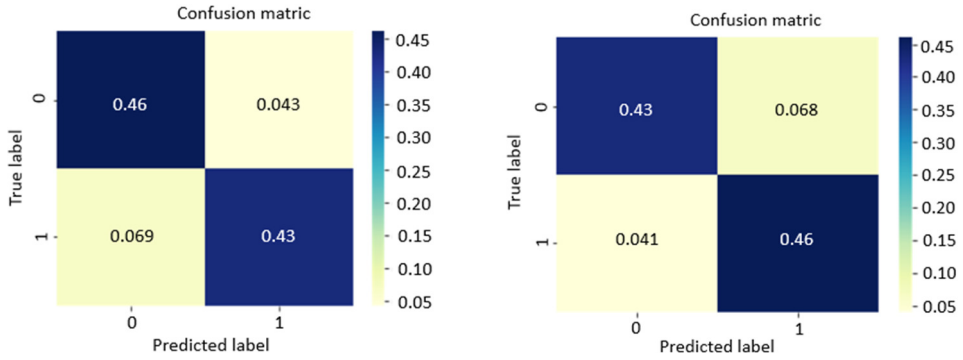



(d) HAN + Bi-GRU model




(e) HAN time distributed model

Fig. 4 Accuracy against loss under each epoch of different demonstrated models with GloVe, W2V, and fastText embedding




(a) HAN+ Bi-GRU model (b) HAN time distributed model

Fig. 5 Confusion matrix of the proposed models with GloVe, W2V, and fastText embedding

## 6. Conclusion and Future Work

The manuscript presents a novel architecture TDHAN incorporating a self-AM with BiLSTM and Bi-GRU. The model uses various text pre-processing techniques and embedding word layers on the IMDb dataset using dense CNN, LSTM, and BiLSTM models. The transformer network utilizes the self-attention process to arrange the input sequence into the hidden

layer, and the word encoder expression utilizes a Bi-GRU to learn factual representations of expressions to assess the interdependency between word representations. The sentence-level encoder uses a Bi-GRU to train significant sentence representations, and attention was used to learn critical sentences. The final output of the model is obtained through a dense and Softmax layer, which classifies the text for the SA task. Based on the main discoveries of the research, the following conclusions are summarized.

(1) The results show that the SA classifier TDHAN and Bi-GRU with word embeddings attains a remarkable 99.70% test accuracy; in contrast, TDHAN with word embeddings reaches a 99.41% test accuracy.

(2) The proposed models are also substantially less susceptible to underfitting and overfitting concerns. It is worth mentioning that the suggested model outperformed the majority of best-performed baselines for the IMDb dataset, as conversant in Table 1 and Section 5.

(3) These methods have made a determined effort to enhance the accuracy of evaluations while using fewer parameters, and research has strived to achieve this with a greater degree of ease. The proposed method involves a comparative analysis of DL models and traditional baselines in text-based classification to evaluate and group reviews into positive or negative categories. The outcomes demonstrate a remarkable improvement in the performance of the suggested model by consolidating word embedding with DL.

In future work, to enhance the quality of embedding, researchers may aim to measure the impact of ELMo, Generative Pre-Training (GPT), and BERT on massive corpora. Additionally, one may address the issue of the emojis used in reviews, as they carry genuine feelings. Moreover, the proposed model performance can be evaluated on different datasets and languages, focusing on analyzing the model's behavior concerning low-resource and multi-lingual languages.

## Conflicts of Interest

The authors declare no conflict of interest.

## References

[1] X. Mao, S. Chang, J. Shi, F. Li, and R. Shi, "Sentiment-Aware Word Embedding for Emotion Classification," Applied Sciences, vol. 9, no. 7, article no. 1334, April 2019.

[2] D. Dangi, A. Bhagat, and D. K. Dixit, "Sentiment Analysis of Social Media Data Based on Chaotic Coyote Optimization Algorithm Based Time Weight-Adaboost Support Vector Machine Approach," Concurrency and Computation: Practice and Experience, vol. 34, no. 3, article no. e6581, February 2022.

[3] A. Pimpalkar and R. J. R. Raj, "Evaluation of Tweets for Content Analysis Using Machine Learning Models," 12th International Conference on Computational Intelligence and Communication Networks, pp. 454-459, September 2020.

[4] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," https://arxiv.org/pdf/1301.3781.pdf, January 16, 2013.

[5] W. Zhao, D. Fang, J. Zhang, Y. Zhao, X. Xu, X. Jiang, et al., "An Effective Framework for Semistructured Document Classification via Hierarchical Attention Model," International Journal of Intelligent Systems, vol. 36, no. 9, pp. 5161-5183, September 2021.

[6] A. Pimpalkar and J. R. Raj R, "MBiLSTMGloVe: Embedding GloVe Knowledge into the Corpus Using Multi-Layer BiLSTM Deep Learning Model for Social Media Sentiment Analysis," Expert Systems with Applications, vol. 203, article no. 117581, October 2022.

[7] D. Dessí, M. Dragoni, G. Fenu, M. Marras, and D. R. Recupero, "Deep Learning Adaptation with Word Embeddings for Sentiment Analysis on Online Course Reviews," Deep Learning-Based Approaches for Sentiment Analysis, Singapore: Springer, pp. 57-83, 2020.

[8] L. Li, T. T. Goh, and D. Jin, "How Textual Quality of Online Reviews Affect Classification Performance: A Case of Deep Learning Sentiment Analysis," Neural Computing and Applications, vol. 32, no. 9, pp. 4387-4415, May 2020.

[9] N. D. Derra and D. Baier, "Working in Detail: How LSTM Hyperparameter Selection Influences Sentiment Analysis Results," Archives of Data Science, Series A, vol. 6, no. 1, pp. 1-22, 2020.

[10] R. Pushpakumar, K. S. Sakunthala Prabha, and P. N. Karthikayan, "For Movie Reviews, A Sentiment Analysis Using Long Short Term Memory Networks," Turkish Journal of Computer and Mathematics Education, vol. 12, no. 9, pp. 1758-1766, April 2021.

[11] P. Cen, K. Zhang, and D. Zheng, "Sentiment Analysis Using Deep Learning Approach," Journal on Artificial Intelligence, vol. 2, no. 1, pp. 17-27, July 2020.

[12] G. Liu and J. Guo, "Bidirectional LSTM with Attention Mechanism and Convolutional Layer for Text Classification," Neurocomputing, vol. 337, pp. 325-338, April 2019.

[13] Y. Ma, H. Fan, and C. Zhao, "Feature-Based Fusion Adversarial Recurrent Neural Networks for Text Sentiment Classification," IEEE Access, vol. 7, pp. 132542-132551, 2019.

[14] S. Li, R. Pan, H. Luo, X. Liu, and G. Zhao, "Adaptive Cross-Contextual Word Embedding for Word Polysemy with Unsupervised Topic Modeling," Knowledge-Based Systems, vol. 218, article no. 106827, April 2021.

[15] I. B. Ltaifa, L. Hlaoua, and L. B. Romdhane, "Hybrid Deep Neural Network-Based Text Representation Model to Improve Microblog Retrieval," Cybernetics and Systems, vol. 51, no. 2, pp. 115-139, 2020.

[16] B. Guo, C. Zhang, J. Liu, and X. Ma, "Improving Text Classification with Weighted Word Embeddings via a Multi-Channel TextCNN Model," Neurocomputing, vol. 363, pp. 366-374, October 2019.

[17] B. Jang, I. Kim, and J. W. Kim, "Word2vec Convolutional Neural Networks for Classification of News Articles and Tweets," PLoS ONE, vol. 14, no. 8, article no. e0220976, 2019.

[18] C. Wang, P. Nulty, and D. Lillis, "A Comparative Study on Word Embeddings in Deep Learning for Text Classification," Proceedings of the 4th International Conference on Natural Language Processing and Information Retrieval, pp. 37-46, December 2020.

[19] M. Kamyab, G. Liu, and M. Adjeisah, "Attention-Based CNN and Bi-LSTM Model Based on TF-IDF and GloVe Word Embedding for Sentiment Analysis," Applied Sciences, vol. 11, no. 23, article no. 11255, December 2021.

[20] M. Cho, J. Ha, C. Park, and S. Park, "Combinatorial Feature Embedding Based on CNN and LSTM for Biomedical Named Entity Recognition," Journal of Biomedical Informatics, vol. 103, article no. 103381, March 2020.

[21] P. Wu, X. Li, L. Chen, S. Ding, and S. Shen, "Sentiment Classification Using Attention Mechanism and Bidirectional Long Short-Term Memory Network," Applied Soft Computing, vol. 112, article no. 107792, November 2021.

[22] S. B. Akbar, K. Thanupillai, and S. Sundararaj, "Combining the Advantages of AlexNet Convolutional Deep Neural Network Optimized with Anopheles Search Algorithm based Feature Extraction and Random Forest Classifier for COVID-19 Classification," Concurrency and Computation: Practice and Experience, vol. 34, no. 15, article no. e6958, July 2022.

[23] A. T. Kabakus, "A Novel COVID-19 Sentiment Analysis in Turkish Based on the Combination of Convolutional Neural Network and Bidirectional Long-Short Term Memory on Twitter," Concurrency and Computation: Practice and Experience, vol. 34, no. 22, article no. e6883, October 2022.

[24] G. Kang, Y. Xiao, J. Liu, Y. Cao, B. Cao, X. Zhang, et al., "Tatt-BiLSTM: Web Service Classification with Topical Attention-Based BiLSTM," Concurrency and Computation: Practice and Experience, vol. 33, no. 16, article no. e6287, August 2021.

[25] J. Wang, Y. Zhang, L. C. Yu, and X. Zhang, "Contextual Sentiment Embeddings via Bi-Directional GRU Language Model," Knowledge-Based Systems, vol. 235, article no. 107663, January 2022.

[26] E. M. Dharma, F. L. Gaol, H. L. H. S. Warnars, and B. Soewito, "The Accuracy Comparison Among Word2vec, Glove, and Fasttext Towards Convolution Neural Network (CNN) Text Classification," Journal of Theoretical and Applied Information Technology, vol. 100, no 2, pp. 349-359, January 2022.

[27] M. Z. Naeem, F. Rustam, A. Mehmood, D. Mui Zzud, I. Ashraf, and G. S. Choi, "Classification of Movie Reviews Using Term Frequency-Inverse Document Frequency and Optimized Machine Learning Algorithms," PeerJ Computer Science, vol. 8, article no. e914, 2022.

[28] N. M. Alharbi, N. S. Alghamdi, E. H. Alkhammash, and J. F. Al Amri, "Evaluation of Sentiment Analysis via Word Embedding and RNN Variants for Amazon Online Reviews," Mathematical Problems in Engineering, vol. 2021, article no. 5536560, September 2021.

[29] M. Bilgin, "A New Statistics-Based Approach to Improve Word2Vec's Sentiment Classification Success," Selcuk University Journal of Engineering Sciences, vol. 20, no. 3, pp. 63-72, 2021.

[30] R. Ranjan, D. Pandey, A. K. Rai, P. Singh, A. Vidyarthi, D. Gupta, et al., "A Manifold-Level Hybrid Deep Learning Approach for Sentiment Classification Using an Autoregressive Model," Applied Sciences, vol. 13, no. 5, article no. 3091, March 2023.