An Efficient Two-State Feature Attention-Based GRU for Text Classification

Dhurgham Ali Mohammed[1,2], Kalyani A. Patel[3]
[1]Faculty of Computer Science, Gujarat University, Ahmedabad, India
[2]Department of Computer Science, Faculty of Education for Girl, University of Kufa, Iraq
[3]K.S. School of Business Management and Information Technology, Gujarat University
E-mail: dhurghama.alhasani@uokufa.edu.iq, kalyanipatel@gujaratuniversity.ac.in

Keywords: Recurrent neural network, gated recurrent unit, two state GRU, feature attention strategy, text classification, deep learning

Received: 1 November, 2024

Text classification has become crucial for mechanically sorting documents into specific categories. The goal of classification is to assign a predefined group or class to an instance based on its characteristics. To attain precise text categorization, a feature selection scheme is employed to categorize significant features and eliminate irrelevant, undesirable, and noisy ones, thereby reducing the dimensionality of the feature space. Many advanced deep learning algorithms have been developed to handle text classification drawbacks. Recurrent neural networks (RNNs) are broadly employed in text classification tasks. In this paper, we referred to a novel Two-state GRU based on a Feature Attention strategy, known as Two-State Feature Attention GRU (TS-FA-GRU). The proposed framework identifies and categorizes word polarity through consecutive mechanisms and word-feature capture. Furthermore, the developed study incorporates a pre-feature attention TS-FA-GRU to capture essential features at an early stage, followed by a post-feature attention GRU that mimics the decoder's function to refine the extracted features. To enhance computational performance, the reset gate in the ordinary GRU is replaced with an update gate, which helps to reduce redundancy and complexity. The effectiveness of the developed model was tested on five benchmark text datasets and compared with five well-established traditional text classification methods. The proposed TS-FA-GRU model demonstrated superior performance over several traditional approaches regarding convergence rate and accuracy. Experimental outcomes revealed that the TS-FA-GRU model achieved excellent text classification accuracies of 93.86%, 92.69%, 94.73%, 92.46%, and 88.23 on the 20NG, R21578, AG News, IMDB, and Amazon review dataset respectively. Moreover, the results indicated that the proposed model effectively minimized the loss function and captured long-term dependencies, leading to exceptional outcomes when compared to the traditional approaches

Povzetek: Predstavljen je nov model za klasifikacijo besedil, imenovan Two-State Feature Attention GRU (TS-FA-GRU), ki temelji na strategiji pozornosti na značilke. Model izboljšuje standardni GRU z zamenjavo reset vrat z posodobitvenimi vrati in uporabo pred- in po-pozornosti za izboljšanje ekstrakcije značilk.

1 Introduction

With the rapid advancements in computer technology and the internet, a tremendous amount of digital textual data is generated daily. Efficiently and precisely retrieving specific content from this vast information pool has become a common challenge [1]. Textual data is highly dimensional, often containing irrelevant and redundant features that are problematic to manage. The issue of data overload was first identified in the early 1960s. Today, a substantial portion of online information exists as structured and unstructured text, with managing the latter posing an essential issue for large organizations [2]. Machine learning delivers a solution by automatically analyzing data, identifying patterns, and making classifications with minimal human intervention. Extracting valuable information relevant to specific interests from a continuously growing pool of documents has become a critical task in machine learning [3]. This necessitates the use of well-organized classification methods capable of assigning texts to one or more classes (labels). These methods have been effectively utilized in many Natural Language Processing (NLP) applications, including sentence classification [4], topic modeling and text clustering [5], spam detection [6], website categorization [7], disease report identification [8], and document summarization [9]. Text classification using RNNs influences the network's capability to process sequences data, making it particularly effective for tasks connecting textual information. RNNs, and their more progressive variants like Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU), are designed to hold information from previous steps in the sequence, allowing them to understand context and dependencies in text [10, 11]. This capability is crucial for tasks such as sentence analysis, language translation, and spam detection, where the meaning of a word often depends on its context within a sentence or document. In recent years,

the rapid increase in text data has made NLP a fascinating area for deep learning tasks. Figure 1 outlines the key steps of NLP, which are utilized across numerous NLP applications. The use of RNNs in text categorization has led to important developments in NLP, making it possible to develop more sophisticated and precise models for understanding and organizing textual data [12].

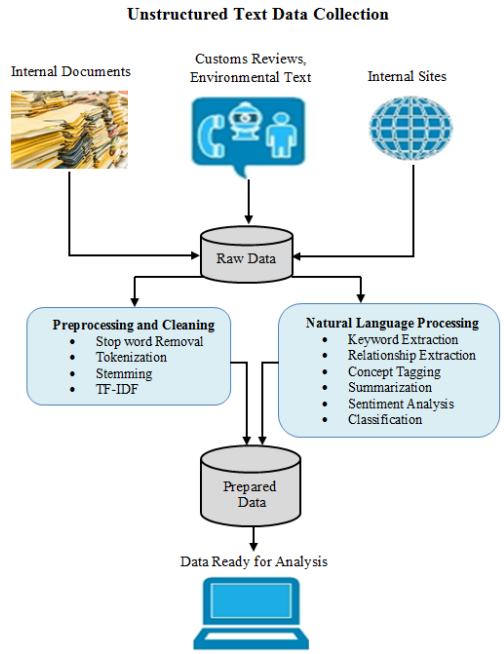

Figure 1: Text classification steps

In recent years, RNNs have been extensively applied in several data mining applications, demonstrating superior performance in classification tasks [13]. RNNs provide excellent semantic composition procedures for text categorization and can capture sequential dependencies in progressive data. There are two main types of RNNs: LSTM networks and GRUs. Text data is high-dimensional with numerous features, making extracting informative features from raw data with a single layer challenging. Therefore, recent deep learning research has focused on utilizing multiple layers to extract highly informative features. Among RNNs, GRUs have shown particular promise in addressing various text classification problems. While previous research has explored the accuracy and performance of GRUs for text classification, certain drawbacks remain in the standard GRU model that require further improvement. Lee et al. [14] established the Word2Vec word embedding technique, which converts each word into a sparse vector corresponding to specific terms, distributing various semantic and syntactic features across each dimension in vector space. Zulqarnain et al. [15] introduced an effective GRU network-based word embedding strategy for sentence classification, which utilized the word2vec technique. Soni et al. [16] utilized Convolutional Neural Networks (CNNs) for sentence modeling, achieving superior text classification results by using ConvoNet methodology to competently categorize sentiment polarity based on the skip-gram method from different positions in a sentence. However, CNNs focus on local features and exclude sequence information, which excels in progressive correlations amongst context and sentiment words through a useful gating mechanism [17]. Despite this, RNNs are still subject to the vanishing gradient and exploding gradient deficiencies [18]. Several deep learning methodologies have been introduced to address these drawbacks, including LSTM [19] and GRU [20]. GRU, an advanced version of LSTM, has been widely employed in numerous NLP applications due to its more effective computational process while retaining the benefits of LSTM. In this study, we examine the effectiveness of the conventional GRU for text classification employing distributed representation on a social platform. The GRU network's capability to tackle the vanishing gradient and exploding gradient issues in typical RNNs is one of its key strengths. We conducted experiments on five benchmark datasets: 20newsgroup (20NG), Reuters21578 (R21578), AG's News, IMDB, and Amazon reviews. Our research aims to improve the typical GRU structure by integrating a Two-State Feature Attention GRU (TS-FA-GRU) model. This model employs an attention mechanism to identify and utilize the most informative features for text classification. The key goal of our study is to boost text classification accuracy while minimizing information loss within the GRU framework. The key contributions of this research are included as follows:

- Introduced a novel the Two-State Feature Attention GRU (TS-FA-GRU) architecture to tackle text classification challenges. This architecture utilizes a feature attention strategy to extract informative features.
- Employed the extensively used unsupervised word embedding technique, GloVe, for vector initialization.
- This study evaluates the impact of replacing the reset gate with an update gate in the candidate state of the traditional GRU network, similar to the approach by Zhou (2016), which finds the absence of the reset gate does not substantially impact on model performance.
- Our contribution focused on developing a mechanism that provides efficient computation and robust performance with fewer parameters.
- This study demonstrates through experimental results that the developed TS-FA-GRU framework performs effectively across benchmark datasets such as 20NG, R21578, AG's News, IMDB, and Amazon review, highlighting its ability to capture long-term dependencies and achieve superior results with significantly lower computational costs compared to traditional approaches.

2 Recent review of text classification

Text classification, an essential task in NLP, has seen substantial developments with the advent of deep learning approaches. This review explores recent developments and innovations in text categorization employing deep learning models [21]. RNNs and their variants, mainly LSTM networks, have been extensively utilized for text classification due to their capability to extract sequential dependencies [22][22]. Singh et al. [23] introduced the Hierarchical Attention Network (HAN) for document categorization, which exploits a hierarchical structure and attention mechanism to emphasis on important words and sentences, achieving state-of-the-art results on numerous benchmarks. Cunha et al. [24] referred to the Transformer language model, which depends completely on self-attention mechanisms to capture long-range dependencies, overcoming the deficiencies of RNNs. BERT (Bidirectional Encoder Representations from Transformers), introduced by Nissa et al. [25], further progressive the arena by utilizing a pre-training method on large corpora and fine-tuning the exact tasks, attaining superior performance across various benchmarks. Gu et al. [26] introduced TextGCN, which constructs a document-word graph and influences GNNs to learn embeddings for text classification, representing substantial enhancements over traditional methods. Recent research has also examined hybrid approaches that combine the strengths of various architectures. For example, Ashraf et al. [27] presented a hybrid approach integrating CNNs and RNNs, leveraging CNNs for capturing local patterns and RNNs for sequential dependencies, achieving notable improvements in classification accuracy. Another hybrid model by Wang et al. [28] combined BERT with Graph Neural Networks to utilize both contextual and relational information, resulting in state-of-the-art performance on different datasets. Conventional RNNs are comprised of fully connected layers that process transitions from the input layer to the hidden layer, and from the hidden layer to the output layer through recurrent connections, as illustrated in Figure 2. Pandian et al. [29] described a three-layer structure enhanced with a "context unit" set, where the connections among hidden layer nodes and context layer nodes have fixed weights. RNNs, being a form of deep neural network tailored for sequential data, are known for their high expressiveness [30].

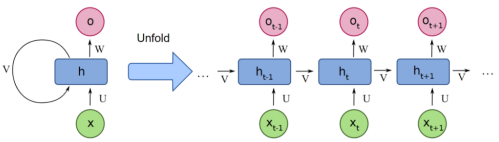

Figure 2: Traditional RNN [31]

RNNs maintain a vector of initiations for each time step, classifying them as a type of DNN. The decision made by an RNN at time step $t-1$ influences the decision at time step t. Therefore, an RNN has two sources of input: the current time step and the preceding one, which determines its response to new data. While standard RNNs can learn patterns in sequential time series data, they are susceptible to the vanishing gradient problem, which affects performance [32]. Despite being powerful for handling sequential data, RNNs are challenging to learn with gradient descent due to the vanishing and exploding gradient issues [33]. These problems are mitigated by advanced variants of standard RNNs, such as LSTM and GRU. GRUs, in particular, have fewer parameters than LSTMs, reducing the risk of overfitting and saving training time. Assumed consecutive input of word vectors (X1, X2, X3...XT), produces a corresponding sequence of hidden states (h1, h2, h3...hT), which is computed at time step t. The output at each step can then be determined using the following RNN calculation:

$$O_t = \varphi(W_x x_t + U_o h_{t-1}) \tag{1}$$

$$H_t^l = \varphi(W_x h_{t-1}^l + U_H h_{t-1}^l) \tag{2}$$

where is the recurrent weights matrix, is the input to-hidden weights matrix, and φ represents an arbitrary activation function. Equations 1 and 2 depict the activity of the hidden layer as influenced by its prior state. In contrast, the GRU, a more efficient and improved version of the LSTM, was initially introduced by Chung et al. [17] for arithmetical machine learning. The GRU, inspired by the LSTM, facilitates information flow within the unit via an update gate z_t and a reset gate r_t, without requiring a distinct memory component [34].

Consequently, the GRU excels in capturing the mapping relationships in time series information whereas giving assistance including minimize complexity and more efficient computation. Figure 3 depicts the structure of the GRU, highlighting the interactions between the update and reset gates.

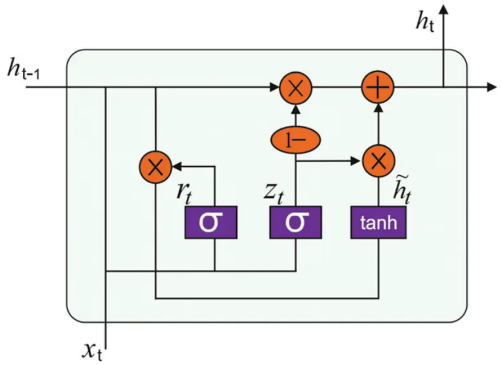

Figure 3: Traditional GRU structure [35]

Like the LSTM, the GRU uses the gating mechanism to regulate the flow of information within the unit, eliminating the need for separate memory cells. The GRU manages and filters information using its internal memory by combining the input and forget gates into a single update gate, which

integrates the previous activation $h_(t-1)$ and the candidate's state h_t. The GRU's three primary components are the update gate, reset gate, and candidate state. The equations governing these mechanisms are presented in equations (3 to 6):

$$z_t = \varphi(V_x z x_t + U_h z h_(t-1) + B_z) \tag{3}$$

$$r_t = \varphi(V_x r x_t + U_h r h_(t-1) + B_r) \tag{4}$$

$$\tilde{h}_t = \tanh(V_{xh\tilde{h}} x_t + U_{hh\tilde{h}}(r_t \cdot h_{t-1}) + B_{\tilde{h}}) \tag{5}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \tag{6}$$

The weight matrices between the input layer and the update gate, the reset gate and the candidate state are indicated as $V_x z$,$V_x r$ and $V_(x\tilde{h})$, respectively, while the recurrent connection weight matrices are indicated by $U_h z$, $U_h r$ and $U_(h\tilde{h})$ respectively. The input sample at time t is represented by x_t, and the hidden state output is denoted as h_t. The sigmoid activation function for the update and reset gates is symbolized by φ, with element-wise multiplication executed by *. The biases corresponding to the update gate, reset gate, and candidate state are illustrated by B_z, B_r and $B_(\tilde{h})$.

Table 1 illustrates the comparison review of various deep learning approaches integrating attention mechanisms for text classification, each model evaluated on datasets such as 20 Newsgroups, Reuters-21578, AG's News, IMDB, and Amazon Reviews, and highlights the model's findings, strengths, and limitations.

3 The proposed framework

In this study, we detail the specific components of the developed architecture, which comprises a two-state GRU based on feature attention strategy GRU, a newly developed model namely, Two-State Feature Attention (TS-FA-GRU). Furthermore, this research investigates the effects of substituting the reset gate r_t with an update gate z_t in the candidate state $\tilde{h}_t$ of traditional GRU design. Consistent with the findings of Zhou et al., [41], our results indicate that the absence of the reset gate does not substantially impact model performance. The established framework utilizes word embeddings as inputs, which are used to extract high-level contextual word features over time steps. The embedding layer predicts these features, then passes them to the two state GRU language mechanism, and ultimately classifies them using a softmax classifier. The primary contribution of the proposed mechanism is its ability to extract crucial features through two main phases: Pre-Feature Attention TS-GRU and Post-Feature Attention TS-GRU.

3.1 Embedding layer

The embedding layer plays a crucial role in developing networks for text classification. The initial step in this process is pre-processing, which entails cleaning the text data. To convert each word in the sentences into a real-valued vector, we employed the pre-trained 300-dimensional GloVe method. These pre-trained vectors adeptly extract both semantic and syntactic information, making them essential for text categorization by transforming word contexts into real-valued feature vectors. Let $L \in R^{V*d}$ determine the embedding inquiry table generated by GloVe, where d is the dimensionality of the words and the vocabulary size denoted by V. Consider a serial of input containing n words and the sentiment resource containing m words. The sequential inputs of the text contexts extract word vectors from L, producing a list of vectors $[W_1, W_2 \ldots, W_n]$ where each $W_i \in R^d$ is the word vector for the corresponding word. Likewise, the sentiment resource sequence extracts word vectors, producing a list $[W^s{}_1, W^s{}_2, \ldots, W^s{}_m]$. This process enables the creation of a matrix for context words and a matrix $W^c = [W_1, W_2 \ldots, W_n] \in R^{n*d}$ for text classification resource words. This approach allows for the establishment of word-level connections between sentiment words and context words, formatted as a correlation matrix, as illustrated in. Essentially, this process combines all word embeddings within V.

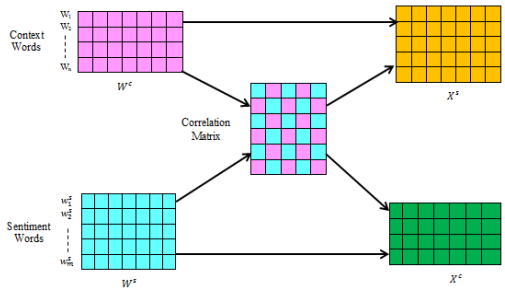

Figure 4: Sentiment-context word correlation

Build _vocab : This function takes the Harvard IV-4 dictionary and text data as input. It outputs a word_frequency array, which includes each word's unique identifier and its frequency in the dataset. Text categorization values range from 0 to 1, where 0 refers to negative analysis, while 1 refers to positive analysis. Words with higher sentiment values have higher word frequency values.

Build_co-occur: This function accepts the word_frequency array and selected text data as inputs, along with parameters for the context window size and minimum count, which are set to 10. The context window size determines how many words are considered to represent the context of each word, and the least count is used to filter out infrequent word cooccurrence pairs. The function determines the range from context_word_id to m_word by counting the words in between. It then generates a sparse matrix containing co-occurrence tuples in the format (word_id, context_word_id, xij), where

Table 1: Comparison summary of various deep learning approaches

Previous Studies	**Approaches**	**Findings**	**Strengths**	**Limitations**
Singh et al. [23]	HAN+Attention Mechanism	Outperformed SOTA approaches in document classification tasks.	Methods documents hierarchically, capturing word and sentence importance.	Complex architecture; requires large datasets for optimal performance.
Ashraf et al. [27]	CNN+RNN	Achieved strong performance by combining CNN for local feature extraction and RNN for capturing sequential patterns.	Captures both local and sequential dependencies effectively.	Increased computational cost and potential over fitting with small datasets.
Nissa et al. [25]	CNN-XGBoost	Obtained SOTA results in different text classification benchmarks.	Handles long-range dependencies and allows for effective parallelization.	Requires extensive computational power and large datasets; may over fit on small datasets.
Wang et al. [28]	BERT + Text FGC	Fine-tuned BERT with Feature Guided Context (FGC) showed significant improvement in text classification tasks, leveraging contextual embeddings.	Provides rich contextual embeddings; achieves state-of-the-art performance on diverse datasets.	Requires large computational resources and extensive training data.
Yao, [36]	Attention-Based BiLSTM	Reported significant enhancement in sentiment analysis tasks.	Captures long-term dependencies and focuses on important words in the text.	Computationally intensive; over fit with limited data.
Liu et al. [37]	CNN with Attention Mechanism	Improved performance in capturing local features for text classification tasks.	Excels at capturing local features and focuses on relevant parts of the text.	They may struggle with capturing long-term dependencies and are sensitive to hyper parameter settings.
Ma et al. [38]	CNN+ Bi-GRU	Combined CNN with Bi-GRU for sequential patterns, improving accuracy and robustness on multiple datasets.	Balances local and bidirectional sequential feature extraction.	Higher complexity individual CNN or RNN models;require careful hyper parameter tuning.
Salini et al. [39]	CNN+ Bi-GRU + Multi Attention Mechanism	Enhanced classification accuracy by incorporating multi-attention mechanisms, allowing the model to dynamically focus on crucial parts of the text.	Integrates multi-attention to prioritize significant features, improving interpret ability and performance.	Increased model complexity and computational demand; risk of over fitting with small datasets.
Guo et al. [40]	Hybrid Models (CNN-RNN-Attention)	Achieved improved accuracy by combining multiple neural network architectures.	Combines strengths of CNNs and RNNs with attention to captured both local and global features.	Increased model complexity; higher computational cost; and potential over fitting with small datasets.

xij represents the co-occurrence value. Figures 5 and 6 demonstrate the flow diagrams for the Build_vocab and Build_cooccur functions, respectively.

Train_GloVe: This function initializes the network parameters and accomplishes the training process. Its usages the co-occurrence data to update the biases and weight vectors during each iteration.

3.2 Removing the reset gate

In the computational framework, the reset and update gates utilize parallel parameters, differing only slightly in their values. The reset gate refers to some complexity and redundancy when interacting with the update gate and candidate state in the GRU model. To address this issue, the proposed approach removes the reset gate r_t from the standard GRU architecture and replaces it with the update gate z_t in the candidate state $\hat{h}_t$. This modification reduces model execution time without significantly affecting accuracy. Consequently, the equations governing the GRU are adjusted as

follows in place of Equations (7) and (8):

Update gate

$$z_t = \varphi(W_{xz}x_t) + U_{hz}(h_{t-1}) + b_z \tag{7}$$

Candidate state

$$\hat{h}_t = \tanh(W_{x\hat{h}}x_t) + U_{h\hat{h}}(r_t \cdot h_{t-1}) + b_{\hat{h}} \tag{8}$$

In this context, the candidate state is denoted by $\hat{h}_t$, with W and b representing the weight and bias respectively. Here, x_t signifies the input, and $h_(t-1)$ indicates the previous time step. By substituting the update gate z_t for the reset gate r_t in the candidate state $\hat{h}_t$, Equation (8) is transformed into Equation (9):

$$\hat{h}_t = \tanh(W_{x\hat{h}}x_t) + U_{h\hat{h}}(z_t \cdot h_{t-1}) + b_{\hat{h}} \tag{9}$$

where z_t is the update gate that replaces the reset gate r_t, and the standard equation for z_t is provided in Equation (23). Additionally, this research substitutes the hyperbolic tangent activation function (tanh) with the Rectified Linear Unit (ReLU) activation function in the candidate state, resulting in the modification of Equation (9) to Equation (10). The contributions are highlighted in red. "In this modification, z_t, the update gate, takes the place of the reset gate r_t, as referred by the ordinary equation in Equation (7). Moreover, this study replaces the tanh with the ReLU in the candidate state$\hat{h}_t$, thereby altering Equation (9) into Equation (10). The specific contributions are emphasized in red."

$$\hat{h}_t = \text{ReLU}(W_{x\hat{h}}x_t) + U_{h\hat{h}}(z_t \cdot h_{t-1}) + b_{\hat{h}} \tag{10}$$

Final output

$$h_t = (1 - z_t) * h_{t-1} + z_t * \hat{h}_t \tag{11}$$

ReLU units have been shown to outperform sigmoid nonlinearities in deep learning algorithms. The final revised architecture of the GRU model is illustrated in Figure 7.

3.3 Two-state GRU strategy

A gated recurrent unit is an advanced type of conventional RNN specifically designed for consecutive modeling. A recurrent layer requires the input value $h_t \in R^n$ at each time step t, as well as the hidden state h_t, by following the recurrent method illustrated in equation (12):

$$h_t = f(W_{xt} + Uh_{t-1} + b) \tag{12}$$

Where $W \in R^{m\times n}$, $b \in R^{m\times m}$, $b \in R^m$, are the weights matrix and bias vector, respectively, and *f* represents the element-wise nonlinearity. Training RNNs to capture long-term dependencies is challenging due to issues like vanishing and exploding gradients [25]. However, by incorporating gating mechanisms, GRUs can retain memory significantly longer than conventional RNNs. Recent studies have revealed that GRUs analyze words employing only the forward language context, making it incredible for them to acquire backward contexts. Consequently, we explored that linguistic processing relies on both forward and backward contexts for accurate sentence interpretation. To address the aforementioned issue, we introduce the Two-State Feature Attention GRU (TS-FA-GRU). The proposed framework employs a two-stage attention strategy to enhance the extraction and utilization of textual features. This framework involves two distinct processes: the "forward pass" for a positive pass and the "backward pass" for a negative pass, as illustrated in Figure 10 . The TS-GRU effectively learns the context of words in both directions. Drawing inspiration from bidirectional recurrent neural networks (BRNNs) described in [42], TS-GRU employs two separate recurrent networks for the forward (left to right) and backward (right to left) passes during training, which are subsequently merged into the output layer. Equations (13 to 16) describe the forward direction of the TS-FA-GRU network, whereas equations (17 to 20) outline the backward direction. These all gates and states such as z_t, r_t, $\tilde{h}_t$, and h_t for both the forward and backward GRU are defined as follows:

Forward Pass:

$$\overrightarrow{z_t} = \sigma(\overrightarrow{W_{zx}}x_t) + \overrightarrow{U_{zh}}(\overrightarrow{h_{t-1}}) + \overrightarrow{b}_z \tag{13}$$

$$\overrightarrow{r_t} = \sigma(\overrightarrow{W_{rx}}x_t) + \overrightarrow{U_{rh}}(\overrightarrow{h_{t-1}}) + \overrightarrow{b_r} \tag{14}$$

$$\overrightarrow{\hat{h}_t} = \tanh(\overrightarrow{W_{\hat{h}}}x_t) + \overrightarrow{r_t} \cdot \overrightarrow{U_{\hat{h}}}(\overrightarrow{h_{t-1}}) + \overrightarrow{b_{\hat{h}}} \tag{15}$$

$$\overrightarrow{h_t} = (1 - \overrightarrow{z_t}) \cdot \overrightarrow{h_{t-1}} + \overrightarrow{z_t} \cdot \overrightarrow{\hat{h}_t} \tag{16}$$

Moreover, we incorporated a backward pass into the developed model to examine further valuable information.

Backward Pass:

$$\overleftarrow{z_t} = \sigma(\overleftarrow{W_{zx}}x_t) + \overleftarrow{U_{zh}}(\overleftarrow{h_{t-1}}) + \overleftarrow{b_z} \tag{17}$$

$$\overleftarrow{r_t} = \sigma(\overleftarrow{W_{rx}}x_t) + \overleftarrow{U_{rh}}(\overleftarrow{h_{t-1}}) + \overleftarrow{b_r} \tag{18}$$

$$\overleftarrow{\hat{h}}_t = \tanh(\overleftarrow{W}_{\hat{h}}x_t) + \overleftarrow{r}_t \cdot \overleftarrow{U_{\hat{h}}}(\overleftarrow{h_{t-1}}) + \overleftarrow{b_{\hat{h}}} \tag{19}$$

$$\overleftarrow{h_t} = (1 - \overleftarrow{z_t}) \cdot \overleftarrow{h_{t-1}} + \overleftarrow{z_t} \cdot \overleftarrow{\hat{h}}_t \tag{20}$$

The activation of a word at time t: donated as $h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}]$ = shows a random sequence (x_1,x_2,…,x_n) consisting n words, where each word at time t is depicted as a spatial vector.

The forward GRU executes $\overrightarrow{(}h_t)$, capturing the left-to-right contexts of the sentence, whereas the backward GRU captures the right-to-left contexts $\overleftarrow{(h_t)}$. These forward and

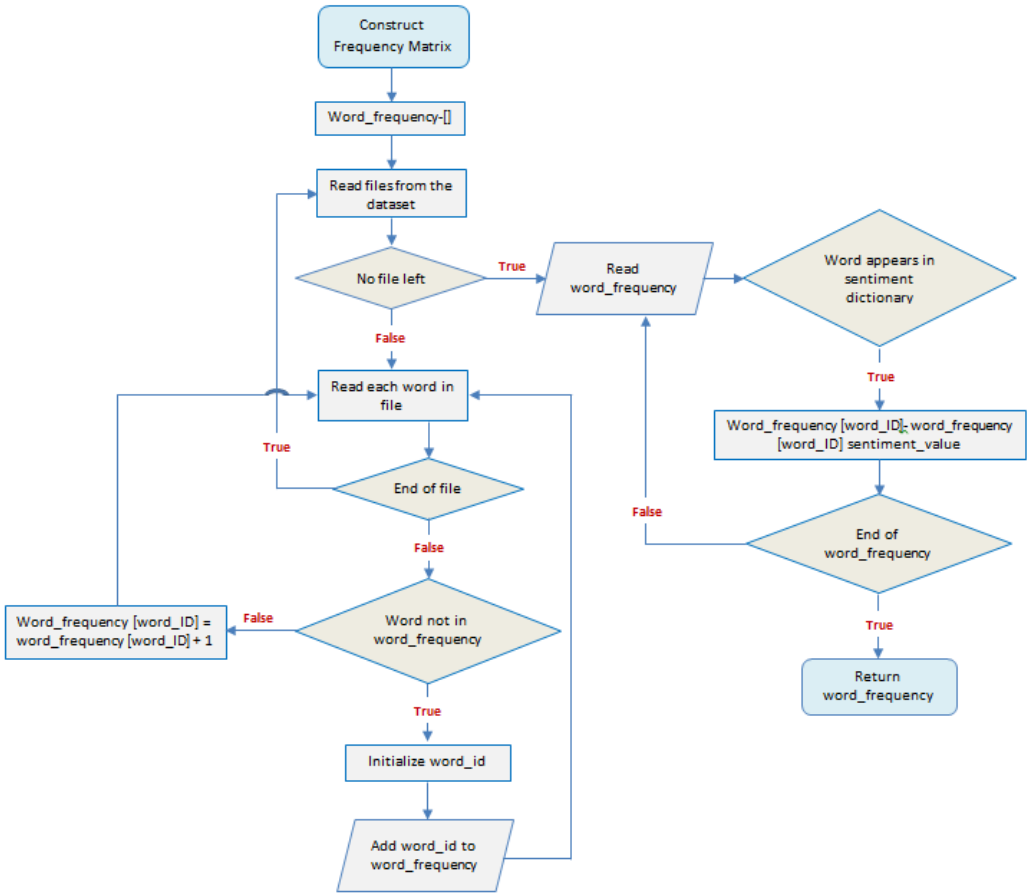

Figure 5: Flowchart illustration of build_vocab function

backward context representations are then joined into a single context. Our proposed mechanism effectively extracts valuable information, which expressively enhances the accuracy of text classification.

3.4 Pre-feature attention TS-GRU

The Pre-feature attention TS-GRU framework is designed to integrate both words and their context to form an initial understanding for sentiment recognition. GRU and LSTM models often struggle to extract essential information for effective text classification, particularly with longer input sequences [43]. However, specific words play a crucial role in enhancing classification accuracy. Within the two-state feature-attention GRU architecture, the attention mechanism is vital for extracting useful information from lengthy reviews [44], aiding in the classification of emotions at the word level. Furthermore, the GRU's gating mechanism regulates the flow of information, and the two-state GRU strategy effectively integrates data from both preceding and succeeding connections [45]. The pre-feature strategy involves both forward and backward sub-states. The forward sub-state processes sequential words from the embedding layer from start to finish, whereas the backward sub-state processes them in reverse order. At any given time step t, for an input word embedding x^k, the forward candidate states $\overrightarrow{h_{t-1}}$ and $\overrightarrow{h_t}$, along with the backward candidate states $\overleftarrow{h_t}$ and $\overleftarrow{h_{t-1}}$, are initialized in the TS-GRU as demonstrated in equations (21, 22, 23):

$$\tilde{h}_t = \tanh(\overrightarrow{W_x^{(\tilde{h})} x_t} + \overrightarrow{r_t} \cdot \overrightarrow{U^{(\tilde{h})} h_{t-1}} + \overrightarrow{b_{\tilde{h}}}) \tag{21}$$

$$\tilde{h}_t = \tanh(\overleftarrow{W_x^{(\tilde{h})} x_t} + \overleftarrow{r_t} \cdot \overleftarrow{U^{(\tilde{h})} h_{t-1}} + \overleftarrow{b_{\tilde{h}}}) \tag{22}$$

$$\bar{h} = \left(V \left[\overrightarrow{\tilde{h}_t} : \overleftarrow{\tilde{h}_t} \right] + k \right) \tag{23}$$

3.5 Attention strategy for word-feature seizing

In the feature-attention process, after obtaining the final output from the first layer's hidden state, we employed an attention strategy that help to identify word polarity by focusing on valuable information in the contextual sentence. The detailed design of the feature attention strategy exploited in our developed approach is shown in Figure 9. Furthermore, Figure 9 demonstrates the dispersal of o_t^k and

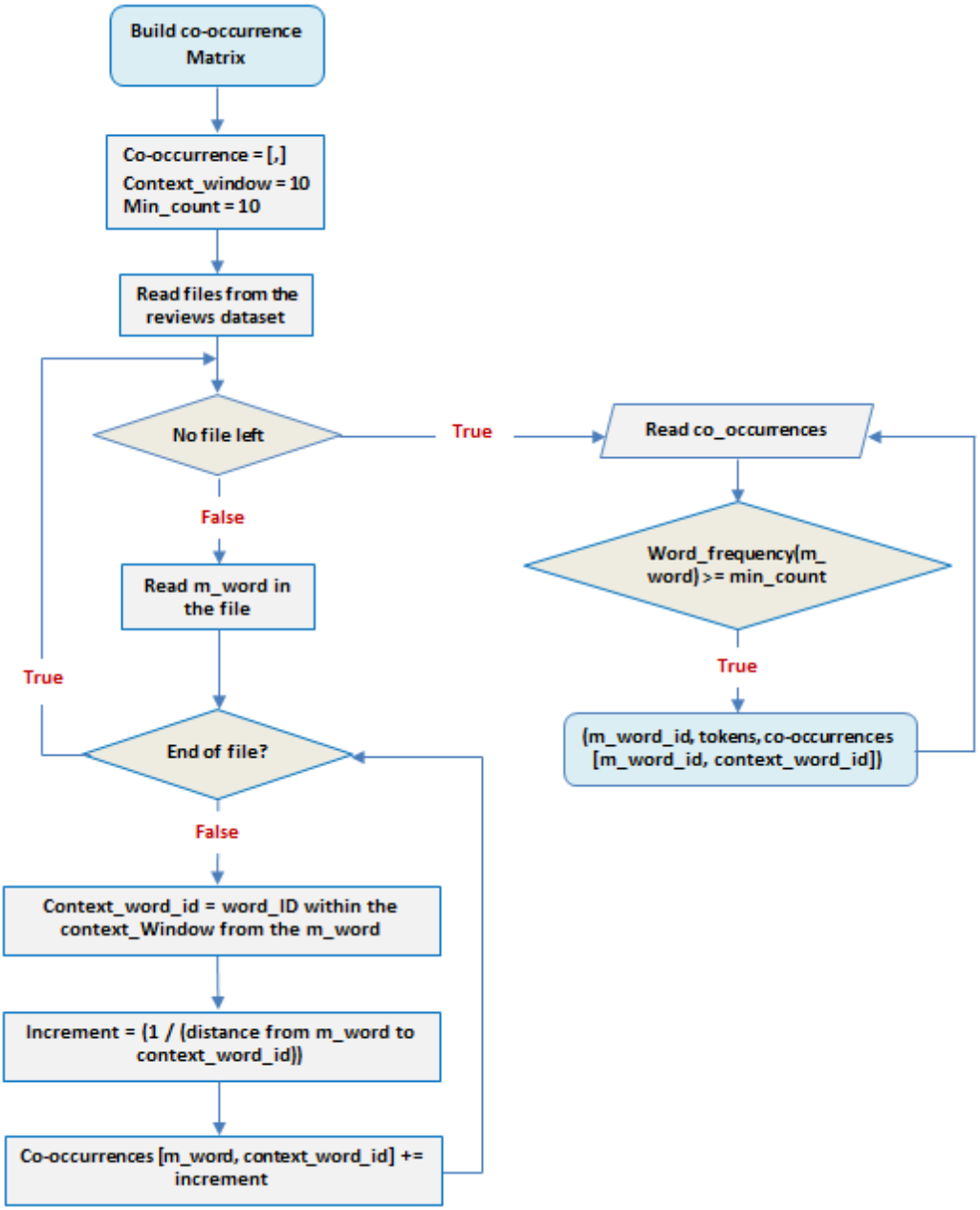

Figure 6: Flowchart illustration of the build cooccurrence function

k^{th} time step, generated by the attention strategy, as follows:

$$o_t^k = \frac{\exp(e_t^k)}{\sum_{i=1}^{m} \exp(e_t^i)} \tag{24}$$

The numerical notation for the memory cell at the e_t^k time step is denoted as k^{th}, and is illustrated in equation 25:

$$e_t^k = [c_{t-1}^T h_1, c_{t-1}^T h_2, \ldots, c_{t-1}^T h_m] \tag{25}$$

Where h_k represents the hidden state of the pre-feature attention TS-GRU, and the memory function at the c_{t-1} at k^{th} time step in the post-feature attention GRU is denoted as c_{t-1}. The target output is then achieved via equation (26) as follows:

$$o_t = \sum_{k=1}^{m} o_t^k h_k \tag{26}$$

3.6 Post-feature attention TS-GRU

Utilizing the feature attention strategy, the post-feature attention mechanism is applied, which further improves these features to gather comprehensive sentence-level information over iterative learning. This stage mimics the human decoding process, where context and relevance are continuously evaluated to improve understanding. The post-

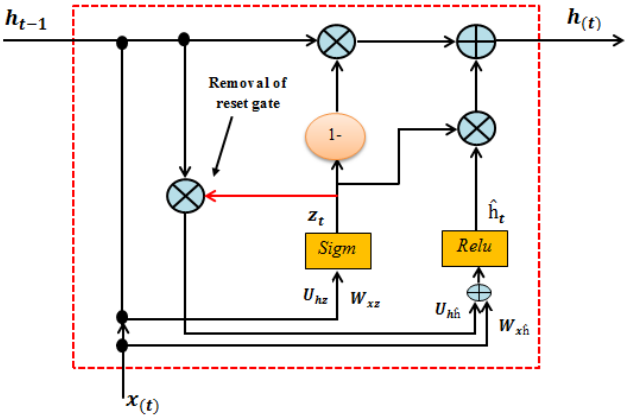

Figure 7: GRU structure excluding the reset gate

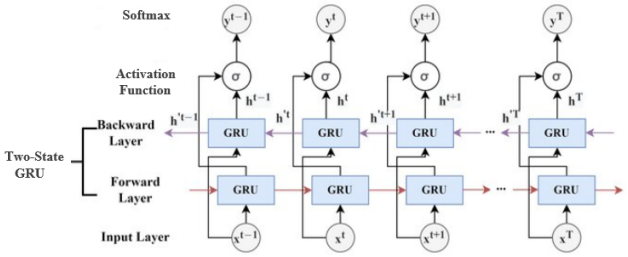

Figure 8: The developed "Two-State GRU strategy for text classification

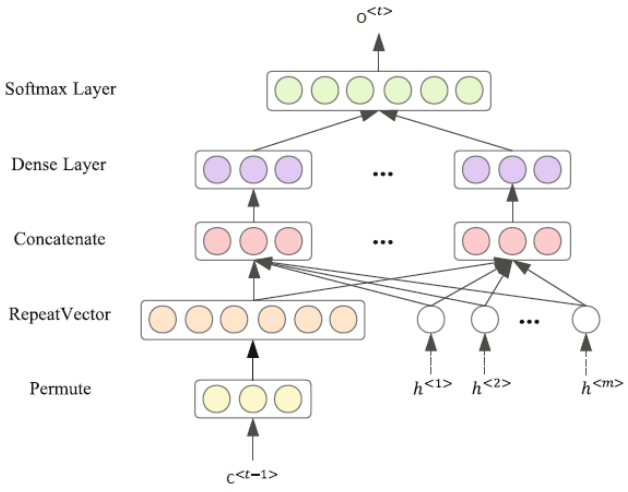

Figure 9: Brief design of feature attention strategy

feature attention TS-GRU integrates the outputs from the initial stage and the attention strategy, ensuring that the developed approach can emphasize vital predictive features. This dual attention strategy allows for Post-Feature Attention TS-GRU which efficiently accomplishes long-term dependencies and intricate patterns in the text, leading to enhanced accuracy in classification tasks. The main equation of the post-feature attention TS-GRU aligns with the typical Bi-GRU, except for the candidate cell, as shown in equation (27):

$$\tilde{h}_t = \tanh(W_x^{(\tilde{h})} x_t + r_t \cdot U^{(\tilde{h})} h_{t-1} + b_{(\tilde{h})}) \quad (27)$$

Thus, the output feature vector from the post-feature attention TS-GRU is passed through a dense layer to serve as the word representation. Finally, a softmax classifier is utilized to predict the class label ("positive" or "negative") for the text classification datasets. It has been noted that extracting and selecting features play a crucial role in enhancing the model's accuracy, as they directly influence its overall performance. Consequently, we introduced the Two-State Feature Attention Two-State GRU (TS-FA-GRU) mechanism specifically for text classification. The comprehensive architecture of the TS-FA-GRU model is demonstrated in Figure 10.

3.7 Developed flowchart

This study presents a systematic approach to text classification, illustrated in the flowchart depicted in Fig. 11. The four-stage process commences with data preprocessing, tailored to text classification objectives. The second stage involves initializing parameters for the developed TS-FA-GRU approach, where an embedding layer transforms words into real-valued vectors, capturing semantic, and syntactic details. Specifically, pre-trained GloVe embeddings are utilized to convert words into vectors, followed by the implementation of two-state GRU integrated with feature attention strategy for extracting more useful features. The third stage monitors training error comparative to a predetermined threshold. Finally, the fourth stage encompasses testing and verification. To assess performance, accuracy, execution time, and error rate are employed as evaluation metrics for text classification tasks. The findings underscore the importance of feature extraction and selection in enhancing model accuracy, as these processes directly impact the model's ultimate performance. To address this, the proposed Feature Attention Two-State GRU mechanism offers an effective solution for sentiment analysis. The findings indicate that effective feature extraction and selection are crucial for enhancing the model's accuracy, as they directly impact its overall performance. To address this, the developed TS-FA-GRU approach has been developed specifically for text classification.

4 Experimental setup

All simulations in this study were conducted on a system with an Intel Core i7-3770 CPU @ 3.40 GHz, equipped with 16 GB of RAM, and operating on Windows 10. Data pre processing and examination were performed using Python 3.9 within the Anaconda development environment, leveraging TensorFlow 1.14 and Keras 2.4 libraries. Moreover, we present a brief overview of our chosen datasets and the hyper parameter configurations used to optimize our proposed model in the following subsection

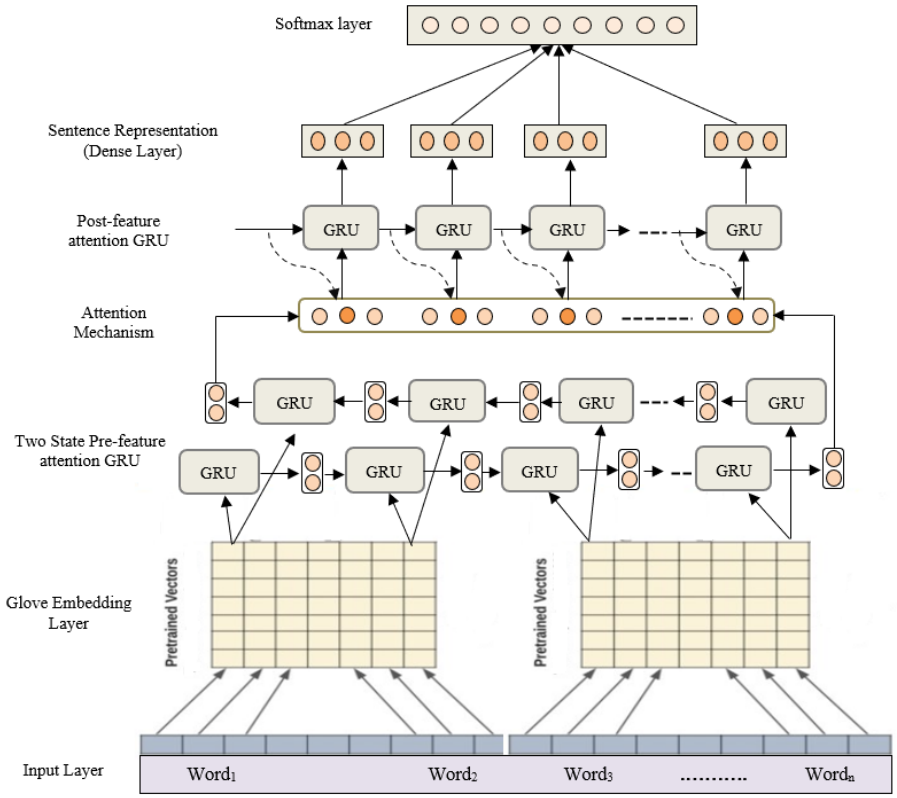

Figure 10: Complete framework of the developed two-state feature attention GRU (TS-FA-GRU)

4.1 Datasets description

In this study, we used five publicly available text classification datasets to evaluate the performance of our proposed model. These text classification datasets included: 20NG, R21578, AG's News, IMDB, and Amazon reviews, and publicly available on https://www.kaggle.com/. To maintain consistency, these datasets were employed for both training and testing in our experiments. This study provides a detailed description of each benchmark dataset used to assess the proposed and traditional approaches. Each dataset represents unique properties and challenges, contributing to a complete evaluation of the model's capabilities. Table 2 refers to a summary of the descriptive statistics for these datasets.

4.2 Data preprocessing steps

Text preprocessing is essential for preparing raw text data for analysis, and plays an important role in the model's performance. The first step in text mining and its applications involves converting unstructured text into a structured format to enhance the quality of the text dataset through preprocessing techniques. These steps are as follows:

Text Cleaning: In this step, we remove unwanted elements from raw text, such as numbers, extra space, and special characters, to make the data more structured. For instance, punctuation and emojis are often removed to emphasize the textual content.

Tokenization: Tokenization splits a text stream into smaller units, such as words, phrases, or other meaningful tokens for easier analysis. The primary goal of tokenization is to analyze and identify the individual words within a sentence.

Stop Word Removal: This stage involves eliminating frequently occurring words that carry little to no meaningful information, filtered out before or after processing natural language data. These common words such as "the," "an," "is," "and," "of," "but," and similar terms. Removing stop words helps emphasize the more meaningful words in the dataset.

Lemmatization: Lemmatization reduces words to their base or root form, considering their grammatical context. For example, "working", "worked", and "works" are converted to "work," preserving the original meaning. Handling Missing Data: Missing text entries are removed or replaced with placeholder values like "Unknown." This ensures the dataset is complete for modeling without introducing biases.

Dimension Reduction: In a text corpus comprising hundreds of thousands of words, it becomes impractical to classify them as features, as it may lead to computational challenges. Therefore, selecting the most representative features is vital to optimize the input for the classification process.

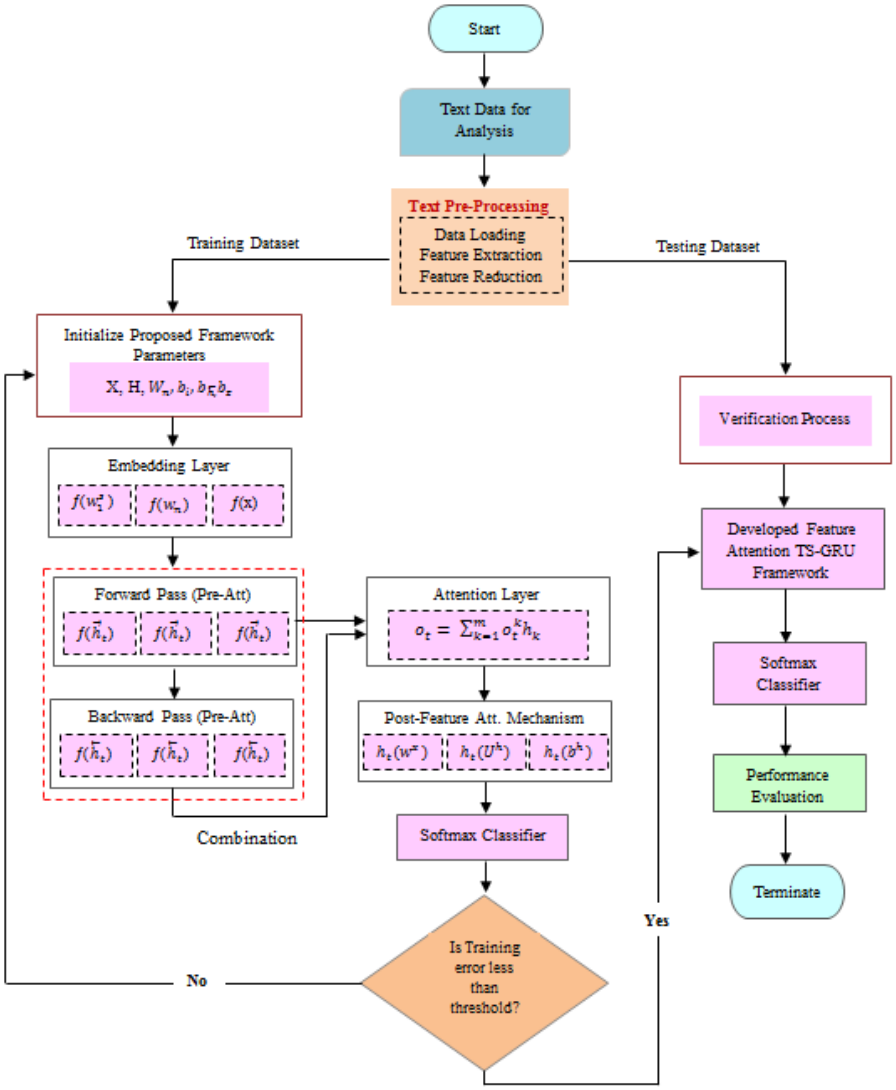

Figure 11: Developed flowchart

Table 2: Descriptive statistics of datasets

Datasets	Train Samples	Test Samples	Average lengths	Classes	Tasks
AG's News	150,000	80,000	42	4	News categorization
R21578	14,600	6400	74	12	Text classification
20NG	10,500	4500	86	15	News classification
IMDB	34,500	15,500	55	2	Sentiment analysis
Amazon Reviews	172,000	77,000	54	2	Sentence classification

4.3 Implementation detail

To optimize the effectiveness of the developed model, we start by improving the quality of the dataset. This study enhances the text dataset by applying pre processing methods, including the removal of stop words (e.g., ”and,” ”our,” ”of,” ”the,” ”to”) and punctuation. We refrain from using stemming during sentence embedding training to preserve all original information. The word embeddings are initialized using 300-dimensional pre-trained GloVe vectors by Pennington et al. [46]. To boost text classification effectiveness, well-ordered training policies for word vectors were employed, as discussed in previous research [47]. A consistent set of embeddings was used to achieve better generalization across datasets based on 30 iterations. During training, the Adam optimizer [48] was utilized with a learning rate of 0.001, and based on available memory we set a mini-batch size of 64. To combat over fitting, a dropout strategy [49] was implemented, including a

dropout rate of 0.5 applied to the output of the TS-GRU layers and a λr coefficient of 10-5 for L2 regularization. Figure 12 provides a configuration summary of each layer in the developed TS-FA-GRU model. The input layer is configured to handle sequences up to 300 words, with longer sentences truncated and shorter ones padded with zeros. The TS-FA-GRU layer comprises 128 memory nodes for both forward and backward passes. To mitigate over fitting, dropout is applied to the GRU layer with a 28% dropout rate for both the input (Dropout-W) and the hidden state (Dropout-U). An additional dropout layer is incorporated after merging the forward and backward GRU layers, with 46% of the input dropped to further reduce over fitting. Finally, a Dense layer was used for sentence illustration, outputting positive or negative predictions. A sigmoid activation function was employed to classify the sentences into two classes, resulting in either a 0 or 1.

4.4 Evaluation matrices

Various evaluation metrics have been used to assess the effectiveness of both the developed approach and conventional approaches in addressing text classification challenges. These metrics include accuracy, precision, recall, F-measure, and error rate/convergence. Precision and recall are among the most frequently utilized assessment metrics, alongside accuracy, for assessing the performance of an approach. Precision measures the accuracy by demonstrating the proportion of correctly classified positive instances among all instances predicted as positive. A higher precision value signifies that the network efficiently classifies true positive cases. Recall, also referred to as sensitivity, signifies the ratio of true positive instances (TP) to the total actual positive cases (TP + FN). The F1 score provides a weighted average of precision and recall, making it a widely used metric for balancing and optimizing a model's performance toward either precision or recall.

$$\text{Accuracy} = \frac{\text{TP}_{(n)}^{(m)} + \text{TN}_{(n)}^{(m)}}{\text{TP}_{(n)}^{(m)} + \text{FP}_{(n)}^{(m)} + \text{FN}_{(n)}^{(m)} + \text{TN}_{(n)}^{(m)}} \quad (28)$$

$$\text{Precision} = \frac{\text{TP}_{(n)}^{(m)}}{\text{TP}_{(n)}^{(m)} + \text{FP}_{(n)}^{(m)}} \quad (29)$$

$$\text{Recall} = \frac{\text{TP}_{(n)}^{(m)}}{\text{TP}_{(n)}^{(m)} + \text{FN}_{(n)}^{(m)}} \quad (30)$$

$$\text{F1 score} = \frac{2 \times \text{Precision}_{(n)}^{(m)} \times \text{Recall}_{(n)}^{(m)}}{\text{Precision}_{(n)}^{(m)} + \text{Recall}_{(n)}^{(m)}} \quad (31)$$

$$\text{Errorrate} = \frac{\text{FP}_{(n)}^{(m)} + \text{FN}_{(n)}^{(m)}}{\text{TP}_{(n)}^{(m)} + \text{TN}_{(n)}^{(m)} + \text{FP}_{(n)}^{(m)} + \text{FN}_{(n)}^{(m)}} \quad (32)$$

5 Results and discussion

In our experimental findings, we present a brief overview of the simulation results for the proposed TS-FA-GRU model, comparing its performance with traditional deep learning models across five different text classification datasets. The evaluation, based on several performance metrics, highlights the significant accuracy achieved by our model for each benchmark dataset.

5.1 Convergence rate

This section describes an experimental evaluation of the developed model with the selected datasets. Figure 17 , 13, 14, 15 and 16 illustrate the learning convergence of the developed and the traditional approaches in text classification tasks. The convergence rate of the proposed TS-FA-GRU model was employed using five benchmark text categorization datasets. We evaluated the convergence rates throughout 30 epochs. It was observed that the error rates for the proposed and comparative algorithms stabilized after 28 epochs. Experimentally, the proposed and comparative approaches demonstrated relatively better convergence on the 20NG, R21578, and IMDB datasets. The AG News and Amazon review datasets are more complex than the 20NG, R21578, and IMDB datasets, causing some interruptions in the convergence rate for all approaches. However, the proposed TS-FA-GRU approach maintained its convergence throughout the 30 epochs, AG News, and Amazon review. It showed slow convergence for all comparative approaches except for the TS-FA-GRU. Moreover, the recurrent approaches, like TS-FA-GRU, Bi-GRU, and Bi-LSTM, performed better on all text classification dataset compared to other traditional approaches including LSTM, GRU, and CNN. The convergence results indicate that the proposed model, incorporating two-state and feature attention mechanism, exhibits excellent performance with faster convergence than the other comparative approaches.

5.2 Accuracy-based analysis

The accuracy-based analysis of the proposed TS-FA-GRU model was also carried out using five benchmark text datasets such as 20NG, R21578, AG News, IMDB, and Amazon review. The evaluation highlighted the model's superior performance, illustrating consistently better accuracy rates across all datasets when compared to traditional deep learning approaches. We evaluated the performance of the developed approach against other methods, including LSTM, GRU, CNN, Bi-LSTM, and Bi-GRU models. Our empirical outcomes showed that the developed approach achieved excellent accuracy across all five datasets compared to these alternative models. These results underscore the effectiveness of the TS-FA-GRU approach in accurately classifying text, indicating its robustness and reliability in various NLP tasks. Table 3 demonstrates the performance

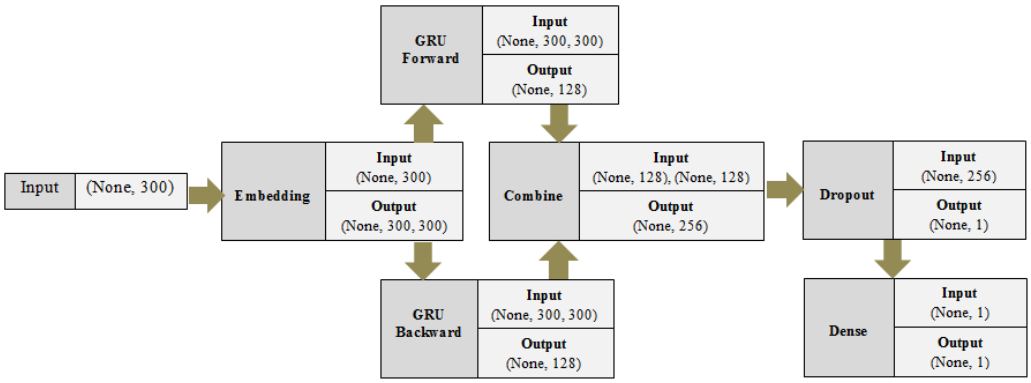

Figure 12: Proposed TS-GRU model configuration

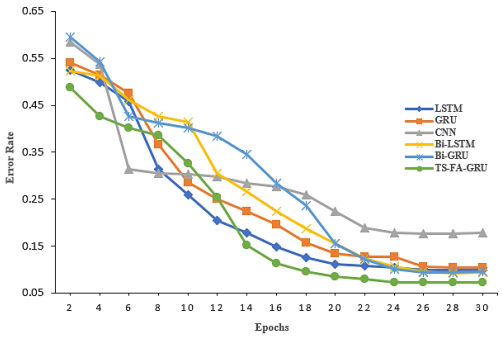

Figure 13: 20NG dataset

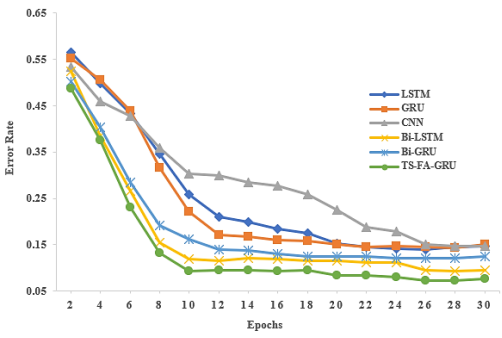

Figure 15: IMDB dataset

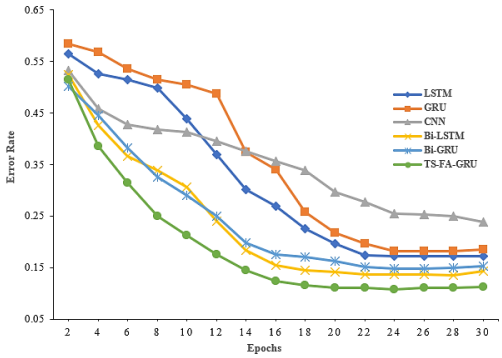

Figure 14: R21578 dataset

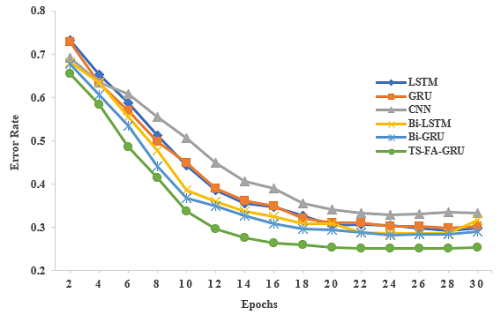

Figure 16: Amazon review dataset

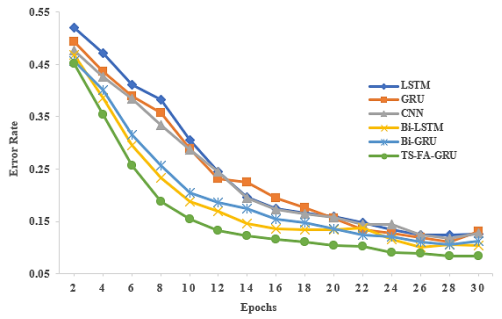

Figure 17: Convergence rate of proposed and comparative models on all respective Fig. 13, 14, 15, 16 datasets

comparison between our developed approaches and traditional approaches across diverse datasets.

5.3 Precision, recall, and F1-score evaluation

The proposed TS-FA-GRU approach was rigorously assessed employing precision, recall, and F1-score metrics across five benchmark text datasets. Precision, which measures the accuracy of positive predictions, showed notable developments with the TS-FA-GRU model compared to conventional approaches. The recall, which evaluates the model's capability to accurately identify all relevant oc-

Table 3: Classification accuracy of proposed and comparative approaches across all distinct datasets

Models	**20NG**	**R21578**	**AG's news**	**IMDB**	**Amazon**
LSTM	89.49	88.41	89.75	86.81	84.27
GRU	90.18	89.01	88.42	86.28	84.92
CNN	88.13	86.36	89.52	85.42	83.97
Bi-LSTM	91.29	90.52	91.45	89.98	85.14
Bi-GRU	90.83	90.18	90.71	89.59	85.79
TS-FA-GRU	93.86	92.69	94.73	92.46	88.23

currences, was also significantly higher in the developed model. This demonstrates its strong capability in capturing the true positives, ensuring that relevant data is not overlooked. Subsequently, the F1-score, which delivers a harmonic mean of precision and recall, further emphasizes the superiority of the TS-FA-GRU model. The higher F1-scores across all datasets highlight the balanced performance of the model. These metrics jointly determine that the proposed approach not only excels in making accurate predictions but also in identifying all relevant instances, offering a comprehensive improvement over conventional deep learning models in text classification tasks. Tables 4,5,6, 7, and 8 display the comparison performance of the developed approach and conventional models. Moreover, as the complexity of the datasets increases, the TS-FA-GRU model demonstrates superior performance than the LSTM, GRU, CNN, Bi-LSTM, and Bi-GRU approaches. In concluded, the proposed model delivered outstanding results, outperforming several conventional approaches in terms of accuracy, particularly on all benchmark datasets.

Table 4: Performance analysis of developed and standard approaches using 20NG dataset

Models	Precision	Recall	F1-score
LSTM	88.92	86.12	87.49
GRU	87.42	88.22	87.94
CNN	85.41	83.65	84.53
Bi-LSTM	91.04	90.16	90.60
Bi-GRU	88.48	90.92	89.64
TS-FA-GRU	92.02	93.18	92.60

Table 5: Performance analysis of developed and traditional models using the R21578 dataset

Models	Precision	Recall	F1-score
LSTM	87.08	88.45	87.74
GRU	86.65	87.68	86.98
CNN	84.60	85.58	84.96
Bi-LSTM	86.54	87.24	86.88
Bi-GRU	88.32	86.56	87.42
TS-FA-GRU	91.72	90.17	90.93

Table 6: Performance analysis of developed and traditional models using AG New's dataset

Models	Precision	Recall	F1-score
LSTM	87.68	85.92	86.56
GRU	89.62	85.84	87.32
CNN	86.31	89.16	87.72
Bi-LSTM	91.35	89.29	90.28
Bi-GRU	90.76	87.82	89.27
TS-FA-GRU	94.04	92.28	93.16

Table 7: Performance analysis of developed and traditional models using IMDB dataset

Models	Precision	Recall	F1-score
LSTM	87.22	86.65	86.93
GRU	88.16	86.42	87.15
CNN	87.74	85.24	86.58
Bi-LSTM	89.52	87.66	88.48
Bi-GRU	90.89	90.42	90.65
TS-FA-GRU	93.24	91.16	92.19

Table 8: Performance analysis of developed and traditional models using Amazon review dataset

Models	Precision	Recall	F1-score
LSTM	79.52	77.28	78.12
GRU	80.77	79.45	79.86
CNN	82.93	81.63	82.27
Bi-LSTM	81.86	84.29	83.05
Bi-GRU	83.19	82.38	82.79
TS-FA-GRU	86.32	84.48	85.40

5.4 Execution time comparison

This study evaluated the execution time performance of the developed model and compared it with standard approaches utilized in all datasets. The computational efficiency of all models depends on the hardware, software, and compiler configurations. To ensure a fair comparison, this research employed the same software setup and hardware combi-

nation for the developed and comparative models. Table 9 illustrates a comparative analysis of the execution times for the developed TS-FA-GRU approach and other traditional approaches, including standard LSTM, GRU, CNN, Bi-LSTM, and Bi-GRU. On the first two datasets, 20NG and R21578, the traditional GRU approach exhibited comparatively better execution times than the developed TS-FA-GRU model due to the simplicity of the data. However, as the complexity and noise levels increase in datasets such as IMDB, AG news, and Amazon review, the developed approach demonstrates superior execution time performance compared to standard LSTM, GRU, CNN, B-LSTM, and Bi-GRU. The best results from the experiments are highlighted in bold.

5.5 Comparison analysis with traditional studies

This study assesses the efficacy of the developed approach by contrasting it with traditional studies, such as cited studies in [50], [51]. The comparison emphasizes on performance in terms of accuracy of the proposed approach using four different datasets: 20NG, AG News, Amazon review, and IMDB. Table 10 shown that our proposed approach consistently outperformed than traditional approaches in terms of accuracy, particularly as the dataset size increases. This enhancement is due to the better generation of textual features, which effectively expands the dataset and diminishes the significance of the initial dataset size. Moreover, our develop approach builds upon the strengths of earlier character-level methods, simplifying the implementation of various languages by openly updating the alphabet. All the experimental evaluations demonstrated that the TS-FA-GRU model consistently converged faster than existing deep learning models, showcasing its competence and effectiveness in attaining optimal performance across diverse datasets.

6 Conclusion and future direction

Text classification is a significant and broadly studied area in NLP. Among the different deep learning models used in NLP, the GRU is notably effective for sequential learning tasks. In this research, we proposed the Two-State Feature Attention GRU (TS-FA-GRU) model to demonstrate a significant improvement in text classification tasks. Our proposed approach leverages the word embedding layer's abilities to examine word polarity through sentential patterns and predict the sentiment in reviews. This research makes three key contributions: Firstly, we introduce the Two-State GRU (TS-GRU) structure to tackle text classification challenges. Secondly, we develop a novel sophisticated feature-attention mechanism that allows the model to dynamically focus on essential features, enhancing its ability to capture intricate dependencies and contextual information within the text through pre- and post-feature attention layers. Moreover, a post-feature attention GRU is employed to mimic the decoder's function, extracting targeted features acquired from the pre-feature attention TS-GRU and the attention layer. Thirdly, this research modifies the standard GRU by removing the reset gate and replacing it with an update gate in the candidate state. Furthermore, we utilized the ReLU activation function in the candidate state of the GRU network instead of the tanh activation function, while softmax is employed as the final output layer for text classification. Through comprehensive evaluations, we conducted experiments using five benchmark datasets such as 20NG, R21578, IMDB, AG News, and Amazon review, the proposed TS-FA-GRU model consistently demonstrated superior accuracy, precision, recall, and F1-score compared to traditional models such as LSTM, GRU, CNN, Bi-LSTM, and Bi-GRU. For future direction, we identified the computational complexity of the proposed model, and attributed the two-state strategy, as an area for improvement. Reducing this complexity throughout the model's processes is a key direction for future research to enhance the model's efficiency. Additionally, we aim to design a more efficient and versatile attention architecture at the word-feature level while also minimizing the overall execution time and computational cost of the developed framework. Additionally, our goal is to create a more efficient and flexible attention structure at the word-feature level, while also minimizing the overall computational cost of the framework we've developed.

References

[1] J. M. Duarte and L. Berton, "A review of semi-supervised learning for text classification," *Artificial intelligence review*, vol. 56, pp. 9401–9469, 2023.

[2] M. Umer, Z. Imtiaz, M. Ahmad, M. Nappi, C. Medaglia, G. S. Choi, and A. Mehmood, "Impact of convolutional neural network and fasttext embedding on text classification," *Multimedia Tools and Applications*, vol. 82, pp. 5569–5585, 2023.

[3] M. Qorib, T. Oladunni, M. Denis, E. Ososanya, and P. Cotae, "Covid-19 vaccine hesitancy: Text mining, sentiment analysis and machine learning on covid-19 vaccination twitter dataset," *Expert Systems with Applications*, vol. 212, p. 118715, 2023.

[4] W. Etaiwi, D. Suleiman, and A. Awajan, "Deep learning based techniques for sentiment analysis: A survey," *Informatica*, vol. 45, no. 7, 2021.

[5] M.-Y. Cheng, D. Kusoemo, and R. A. Gosno, "Text mining-based construction site accident classification using hybrid supervised machine learning," *Automation in Construction*, vol. 118, p. 103265, 2020.

[6] M. Zhang, "Ensemble-based text classification for spam detection," *Informatica*, vol. 48, no. 6, 2024.

Table 9: Training times for different models on various datasets

Models	20NG	R21578	AG news	IMDB	Amazon
LSTM	12m 03s	13m 21s	39m 45s	27m 50s	50m 28s
GRU	11m 43s	10m 32s	34m 16s	22m 43s	45m 30s
CNN	19m 22s	17m 52s	49m 37s	39m 08s	57m 22s
Bi-LSTM	17m 24s	19m 48s	55m 16s	40m 19s	1 h 14m 24s
Bi-GRU	16m 55s	17m 28s	48m 24s	34m 13s	1 h 02m 21s
TS-FA-GRU	14m 40s	15m 26s	40m 36s	31m 51s	55m 04s

Table 10: Comparison analysis with traditional studies in terms of accuracy

Methods	20NG	AG's news	Amazon	IMDB
ROBERTA+ULR [50]	58.19	85.19	—	—
SWEM-max [52]	—	91.80	—	—
SWEM-hier [52]	—	92.48	—	—
Generative LSTM[53]	—	90.70	—	—
Generative LSTM-Shard comp.[53]	—	90.60	—	—
OOD Methods [54]	87.44	—-	—	—
FastText (Word Matrix) [55]	82.04	90.18	—	—
FastText (Document Matrix) [55]	82.28	90.05	—	—
CCNN-GAN [56]	88.14	91.94	—	—
PV+GRU [55]	—	—	81.82	—
NAE-GRU [56]	—	—	87.43	—
CNN+LSTM [57]	—	—	88.20	—
XLNet-Large (ensemble) [58]	—	—	67.64	—
CNN+GRU [59]	—	—	87.50	—
FARNN-Att [57]	—	—	—	89.22
WALE-LSTM [58]	—	—	—	89.50
CNN-LSTM [59]	—	—	—	88.90
CBOW-D+CNN [60]	—	—	—	87.20
TS-FA-GRU (Proposed)	93.86	94.73	88.23	92.46

[7] L. Tang and Q. H. Mahmoud, "A survey of machine learning-based solutions for phishing website detection," *Machine Learning and Knowledge Extraction*, vol. 3, pp. 672–694, 2021.

[8] J. P. Li, A. U. Haq, S. U. Din, J. Khan, A. Khan, and A. Saboor, "Heart disease identification method using machine learning classification in e-healthcare," *IEEE access*, vol. 8, pp. 107562–107582, 2020.

[9] Y. Fang and Y. Wang, "Cross-modal sentiment analysis of text image fusion based on hybrid fusion strategy," *Informatica*, vol. 48, no. 21, 2024.

[10] M. Zulqarnain, R. Ghazali, Y. M. M. Hassim, and M. Rehan, "A comparative review on deep learning models for text classification," *Indones. J. Electr. Eng. Comput. Sci*, vol. 19, pp. 325–335, 2020.

[11] D. S. Asudani, N. K. Nagwani, and P. Singh, "Impact of word embedding models on text analytics in deep learning environment: a review," *Artificial intelligence review*, vol. 56, pp. 10345–10425, 2023.

[12] K. Wang, Y. Ding, and S. C. Han, "Graph neural networks for text classification: A survey," *Artificial Intelligence Review*, vol. 57, p. 190, 2024.

[13] A. Mariyam, S. A. H. Basha, and S. V. Raju, "A literature survey on recurrent attention learning for text classification," in *IOP Conference Series: Materials Science and Engineering*, vol. 1042, p. 012030, IOP Publishing, 2021.

[14] O.-J. Lee and J. J. Jung, "Story embedding: Learning distributed representations of stories based on character networks," *Artificial Intelligence*, vol. 281, p. 103235, 2020.

[15] M. Zulqarnain, R. Ghazali, M. G. Ghouse, and M. F. Mushtaq, "Efficient processing of gru based on word embedding for text classification," *JOIV: International Journal on Informatics Visualization*, vol. 3, pp. 377–383, 2019.

[16] S. Soni, S. S. Chouhan, and S. S. Rathore, "Textconvonet: A convolutional neural network based archi-

tecture for text classification," *Applied Intelligence*, vol. 53, no. 11, pp. 14249–14268, 2023.

[17] Y. Huang, X. Dai, J. Yu, and Z. Huang, "Sa-sgru: combining improved self-attention and skip-gru for text classification," *Applied Sciences*, vol. 13, no. 3, p. 1296, 2023.

[18] S. Bo, Y. Zhang, J. Huang, S. Liu, Z. Chen, and Z. Li, "Attention mechanism and context modeling system for text mining machine translation," in *2024 6th International Conference on Data-driven Optimization of Complex Systems (DOCS)*, pp. 857–863, IEEE, 2024.

[19] R. C. Staudemeyer and E. R. Morris, "Understanding lstm–a tutorial into long short-term memory recurrent neural networks," *arXiv preprint arXiv:1909.09586*, 2019.

[20] K. Cho, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[21] W. Ahmad, H. U. Khan, T. Iqbal, and S. Iqbal, "Attention-based multi-channel gated recurrent neural networks: a novel feature-centric approach for aspect-based sentiment classification," *IEEE Access*, vol. 11, pp. 54408–54427, 2023.

[22] N. Venkateswaran, R. Vidhya, D. A. Naik, T. F. M. Raj, N. Munjal, and S. Boopathi, *Study on Sentence and Question Formation Using Deep Learning Techniques*, pp. 252–273. IGI Global, 2023.

[23] G. Singh, A. Nagpal, and V. Singh, "Optimal feature selection and invasive weed tunicate swarm algorithm-based hierarchical attention network for text classification," *Connection Science*, vol. 35, p. 2231171, 2023.

[24] W. Cunha, F. Viegas, C. França, T. Rosa, L. Rocha, and M. A. Gonçalves, "A comparative survey of instance selection methods applied to non-neural and transformer-based text classification," *ACM Computing Surveys*, vol. 55, pp. 1–52, 2023.

[25] N. K. Nissa and E. Yulianti, "Multi-label text classification of indonesian customer reviews using bidirectional encoder representations from transformers language model," *Int. J. Power Electron. Drive Syst*, vol. 13, pp. 5641–5652, 2023.

[26] Y. Gu, Y. Wang, H.-R. Zhang, J. Wu, and X. Gu, "Enhancing text classification by graph neural networks with multi-granular topic-aware graph," *IEEE Access*, vol. 11, pp. 20169–20183, 2023.

[27] M. Ashraf, F. Abid, I. U. Din, J. Rasheed, M. Yesiltepe, S. F. Yeo, and M. T. Ersoy, "A hybrid cnn and rnn variant model for music classification," *Applied Sciences*, vol. 13, p. 1476, 2023.

[28] Y. Wang, C. Wang, J. Zhan, W. Ma, and Y. Jiang, "Text fcg: Fusing contextual information via graph learning for text classification," *Expert Systems with Applications*, vol. 219, p. 119658, 2023.

[29] A. P. Pandian, "Performance evaluation and comparison using deep learning techniques in sentiment analysis," *Journal of Soft Computing Paradigm (JSCP)*, vol. 3, pp. 123–134, 2021.

[30] M. Zulqarnain, R. Ghazali, H. Shah, L. H. Ismail, A. Alsheddy, and M. Mahmud, "A deep two-state gated recurrent unit for particulate matter (pm2. 5) concentration forecasting," *Computers, Materials & Continua*, vol. 71, no. 2, pp. 3051–3068, 2022.

[31] B. Bramantyo, M. Pajar, K. Putra, and N. Hendrastuty, "Implementasi recurrent neural network pada multiclass text classification judul berita," *Jurnal Media Borneo*, vol. 1, 2023.

[32] J. Du, C.-M. Vong, and C. L. P. Chen, "Novel efficient rnn and lstm-like architectures: Recurrent and gated broad learning systems and their applications for text classification," *IEEE transactions on cybernetics*, vol. 51, pp. 1586–1597, 2020.

[33] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning–based text classification: a comprehensive review," *ACM computing surveys (CSUR)*, vol. 54, pp. 1–40, 2021.

[34] M. Zulqarnain, R. Ghazali, S. Khaleefah, and A. Rehan, "An improved the performance of gru model based on batch normalization for sentence classification," *Int. J. Comput. Sci. Netw. Secur*, vol. 19, no. 9, pp. 176–186, 2019.

[35] Y. Fang, S. Yang, B. Zhao, and C. Huang, "Cyberbullying detection in social networks using bi-gru with self-attention mechanism," *Information*, vol. 12, p. 171, 2021.

[36] X. Yao, "Attention-based bilstm neural networks for sentiment classification of short texts," in *Proc. Int. Conf. Inf. Sci. Cloud Comput*, pp. 110–117, 2017.

[37] Z. Liu, H. Huang, C. Lu, and S. Lyu, "Multichannel cnn with attention for text classification," *arXiv preprint arXiv:2006.16174*, 2020.

[38] Y. Ma, H. Chen, Q. Wang, and X. Zheng, "Text classification model based on cnn and bigru fusion attention mechanism," in *ITM Web of Conferences*, vol. 47, p. 02040, EDP Sciences, 2022.

[39] Y. Salini, P. Eswaraiah, M. V. Brahmam, and U. Sirisha, "Word embedding for text classification: Efficient cnn and bi-gru fusion multi attention mechanism," *EAI Endorsed Transactions on Scalable Information Systems*, vol. 10, no. 6, 2023.

[40] L. Guo, D. Zhang, L. Wang, H. Wang, and B. Cui, "Cran: a hybrid cnn-rnn attention-based model for text classification," in *Conceptual Modeling: 37th International Conference, ER 2018, Xi'an, China, October 22–25, 2018, Proceedings 37*, pp. 571–585, Springer, 2018.

[41] G.-B. Zhou, J. Wu, C.-L. Zhang, and Z.-H. Zhou, "Minimal gated unit for recurrent neural networks," *International Journal of Automation and Computing*, vol. 13, pp. 226–234, 2016.

[42] C. Nathwani, "Online signature verification using bidirectional recurrent neural network," in *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 1076–1078, IEEE, 2020.

[43] W. Xu, J. Chen, Z. Ding, and J. Wang, "Text sentiment analysis and classification based on bidirectional gated recurrent units (grus) model," *arXiv preprint arXiv:2404.17123*, 2024.

[44] M. Zulqarnain, S. A. Ishak, R. Ghazali, N. M. Nawi, M. Aamir, and Y. M. M. Hassim, "An improved deep learning approach based on variant two-state gated recurrent unit and word embeddings for sentiment classification," *International Journal of Advanced Computer Science and Applications*, vol. 11, 2020.

[45] A. Bachir Kaddis Beshay, "Cold-start active learning for text classification of business documents," Master's thesis, University of Twente, 2023.

[46] A. A. Metwally, P. S. Yu, D. Reiman, Y. Dai, P. W. Finn, and D. L. Perkins, "Utilizing longitudinal microbiome taxonomic profiles to predict food allergy via long short-term memory networks," *PLoS computational biology*, vol. 15, p. e1006693, 2019.

[47] Y. Hao, Y. Sheng, and J. Wang, "Variant gated recurrent units with encoders to preprocess packets for payload-aware intrusion detection," *IEEE Access*, vol. 7, pp. 49985–49998, 2019.

[48] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[49] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, pp. 1929–1958, 2014.

[50] Z. Chu, K. Stratos, and K. Gimpel, "Unsupervised label refinement improves dataless text classification," *arXiv preprint arXiv:2012.04194*, 2020.

[51] B. Liu, "Text sentiment analysis based on cbow model and deep learning in big data environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, pp. 451–458, 2020.

[52] D. Shen, G. Wang, W. Wang, M. R. Min, Q. Su, Y. Zhang, C. Li, R. Henao, and L. Carin, "Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms," *arXiv preprint arXiv:1805.09843*, 2018.

[53] D. Yogatama, C. Dyer, W. Ling, and P. Blunsom, "Generative and discriminative text classification with recurrent neural networks," *arXiv preprint arXiv:1703.01898*, 2017.

[54] L. Kong, H. Jiang, Y. Zhuang, J. Lyu, T. Zhao, and C. Zhang, "Calibrated language model fine-tuning for in-and out-of-distribution data," *arXiv preprint arXiv:2010.11506*, 2020.

[55] J. Xu and Q. Du, "A deep investigation into fasttext," in *2019 IEEE 21st international conference on high performance computing and communications; IEEE 17th international conference on smart city; IEEE 5th International conference on data science and systems (HPCC/SmartCity/DSS)*, pp. 1714–1719, IEEE, 2019.

[56] T. Wang, L. Liu, H. Zhang, L. Zhang, and X. Chen, "Joint character-level convolutional and generative adversarial networks for text classification," *Complexity*, vol. 2020, no. 1, p. 8516216, 2020.

[57] 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.

[58] X. Fu, J. Yang, J. Li, M. Fang, and H. Wang, "Lexicon-enhanced lstm with attention for general sentiment analysis," *IEEE Access*, vol. 6, pp. 71884–71891, 2018.

[59] J. Camacho-Collados and M. T. Pilehvar, "On the role of text preprocessing in neural network architectures: An evaluation study on text categorization and sentiment analysis," *arXiv preprint arXiv:1707.01780*, 2017.

[60] B. Liu, "Text sentiment analysis based on cbow model and deep learning in big data environment," *Journal of ambient intelligence and humanized computing*, vol. 11, no. 2, pp. 451–458, 2020.