Enhancing Sentiment Analysis on Social Media Data with Advanced Deep Learning Techniques

Huu-Hoa Nguyen
College of Information and Communication Technology, Can Tho University, Vietnam

***Abstract*—This paper introduces a comprehensive methodology for conducting sentiment analysis on social media using advanced deep learning techniques to address the unique challenges of this domain. As digital platforms play an increasingly pivotal role in shaping public discourse, the demand for real-time sentiment analysis has expanded across various sectors, including policymaking, brand monitoring, and personalized services. Our study details a robust framework that encompasses every phase of the deep learning process, from data collection and preprocessing to feature extraction and model optimization. We implement sophisticated data preprocessing techniques to improve data quality and adopt innovative feature extraction methods such as TF-IDF, Word2Vec, and GloVe. Our approach integrates several advanced deep learning configurations, including variants of BiLSTMs, and employs tools like Scikit-learn and Gensim for efficient hyperparameter tuning and model optimization. Through meticulous optimization with GridSearchCV, we enhance the robustness and generalizability of our models. We conduct extensive experimental analysis to evaluate these models against multiple configurations using standard metrics to identify the most effective techniques. Additionally, we benchmark our methods against prior studies, and our findings demonstrate that our proposed approaches outperform comparative techniques. These results provide valuable insights for implementing deep learning in sentiment analysis and contribute to setting benchmarks in the field, thus advancing both the theoretical and practical applications of sentiment analysis in real-world scenarios.**

Keywords*—Sentiment analysis; deep learning; hyperparameter; feature extraction; social media; digital platform; gridsearchcv; BiLSTM; TF-IDF; word2vec; glove; Scikit-learn and Gensim

I. INTRODUCTION

In the digital era, social networks such as Twitter, now known as *X*, play a pivotal role in shaping public discourse and capturing real-time public sentiment [1]. These platforms provide unprecedented access to vast streams of user-generated content, reflecting the collective mood on topics ranging from daily interests to major global events. This rich dataset is fertile ground for sentiment analysis, crucial for understanding social dynamics and applications such as policy-making and personalized services. Automating the classification of sentiment in text data effectively enables stakeholders to respond more swiftly and appropriately to public opinion [2].

Despite its extensive utility, sentiment analysis poses several practical challenges, especially in the context of social media where language use is diverse and constantly evolving [3]. Machine learning, particularly deep learning, has emerged as a robust solution to these complexities. These techniques excel at deciphering subtle nuances of language on social media by modeling high-level abstractions in data [4].

However, deploying deep learning for sentiment analysis involves navigating a range of technical challenges across the deep learning workflow. This includes data acquisition and preprocessing, selection and application of feature extraction techniques, choice and tuning of models, and rigorous analysis of model performance. Each stage is critical; for example, effective data preprocessing significantly reduces noise and enhances the quality of the dataset, while the choice of feature extraction method greatly impacts the model's ability to correctly interpret and classify sentiment [5].

The motivations behind our proposed approach stem from the need to enhance the accuracy and efficiency of sentiment analysis in the ever-evolving landscape of social media. Traditional methods often fall short due to their inability to handle the vast diversity and rapid changes in language use on these platforms. By utilizing advanced deep learning techniques, our approach aims to overcome these limitations, providing a more understanding of public sentiment.

The main contributions in this paper include multifold advancements in sentiment analysis. First, we introduce a comprehensive system architecture that covers all phases of the deep learning process, with careful attention to each stage. This approach integrates advanced data preprocessing strategies and innovative feature extraction methods such as TF-IDF, Word2Vec, and GloVe, utilizing tools from well-known libraries like Scikit-learn and Gensim [21]. Second, we explore the use of advanced deep learning frameworks like BiLSTM, optimizing each model's configuration to maximize performance. Third, we conduct extensive experiments to evaluate and compare these models across various configurations, thoroughly analyzing their performance to identify the most effective approaches for sentiment classification. Lastly, we benchmark our methods against prior studies, helping to establish new standards in the field.

The structure of this paper is organized as follows. Section II explores related works on sentiment analysis. Section III presents the foundations of deep learning. Section IV details our proposed methods. Section V focuses on our experimental analysis and comparison of various models. Finally, Section VI concludes with a summary of findings and future research directions.

II. RELATED WORK

The field of sentiment analysis has witnessed substantial contributions that employ various language processing techniques aimed at refining data to enhance accuracy. One notable approach involves using regular expressions, as demonstrated by the TransRegex tool introduced by [6], which significantly improved accuracy across diverse datasets by removing extraneous elements like special characters, URLs, or HTML tags. Moreover, focused classification and reduction of stop words have been shown to substantially reduce corpus size and enhance overall accuracy [7]. Challenges specific to language, such as addressing spelling errors and the need for word normalization, have been tackled with algorithms like Damerau-Levenshtein [8] and targeted lemmatization techniques, which notably increase accuracy in sentiment analysis for languages like Bangla [9].

The application of linguistic analysis extends beyond everyday communications to encompass political and social domains. Studies such as [10] and [11] have illustrated the effectiveness of preprocessing in improving sentiment analysis outcomes in diverse contexts, including political events and film reviews. Moreover, the role of machine learning in identifying and analysing patterns of hate speech on platforms like Twitter has been examined, with algorithms like Naïve Bayes demonstrating superior performance in detecting and categorizing hateful content [12].

The vast data generated on social networks has been a rich source for sentiment analysis, as exemplified by research focusing on political sentiments during the Jakarta Governorship Election [13]. Here, the use of techniques like TF-IDF [20] for feature extraction and the application of k-fold cross-validation methods underscored the potential for machine learning in improving accuracy in sentiment prediction.

Innovative approaches have also been explored for the deeper analysis of textual data, integrating models such as CNN and LSTM to process large datasets, including movie reviews on platforms like IMDB [14]. These deep learning models have shown remarkable efficacy in classifying sentiments with high accuracy, illustrating the advantage of advanced algorithms in extracting emotional content from text.

Comparative studies have further highlighted the diversity of machine learning and deep learning methods in sentiment analysis tasks. The contrast between classical machine learning techniques and the more complex deep learning approaches, particularly in their methods of converting text into analysable vectors, reveals a spectrum of accuracy and efficiency in sentiment classification [15]. This variety of methodologies highlights the continuous evolution of sentiment analysis, promoting the use of supervised and unsupervised learning models to improve accuracies across different domains.

This landscape of related work reflects the dynamic nature of sentiment analysis research. It also points towards the continuous need for innovation in processing techniques and algorithmic strategies to tackle the complexities of natural language and the reliability of sentiment analysis outcomes.

III. Deep Learning Foundations

Deep learning, a branch of machine learning, employs hierarchical neural networks to model complex patterns and high-level abstractions in data. Unlike traditional machine learning techniques such as Logistic Regression and Support Vector Machine, which are effective for tasks where the relationship between input and output is less intricate, deep learning excels in scenarios where the predictive factors involve complex relationships and high-dimensional data, such as sentiment and emotion classification. Traditional models often require manual feature extraction and selection, whereas deep learning networks automatically learn feature representations from raw data, removing the need for manual intervention.

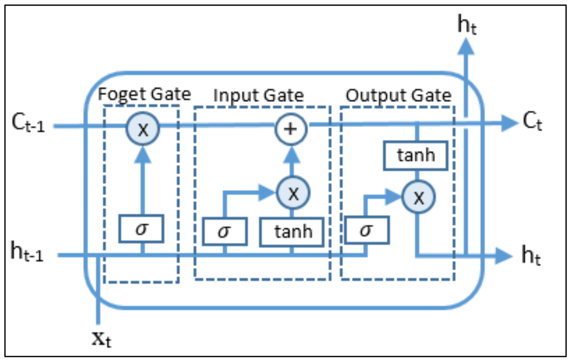


Fig. 1. LSTM architecture.

This section explores several advanced deep learning models such as LSTM, GRU, Bi-LSTM, Bi-GRU, CNN-LSTM, and ConvLSTM. It highlights how these technologies enhance sentiment analysis through predictive modeling.

A. *Long Short-Term Memory (LSTM)*

LSTM networks represent a crucial innovation in neural networks [26]. As an enhancement of Recurrent Neural Networks, LSTMs are adept at recognizing patterns in extended sequences of data, essential for tasks like time series prediction.

As illustrated in Fig. 1, LSTMs consist of interconnected cells featuring three main gates: forget, input, and output. These gates control the flow and modification of information within the network, helping to maintain, update, and retrieve data across different time steps. This selective memory capability significantly enhances decision-making processes.

Focusing on sentiment analysis, the strength of LSTMs lies in their capability to understand the context and nuances over longer text sequences, making them ideal for analyzing opinions and emotions in user-generated content. By utilizing this technology, deep learning models can more accurately gauge sentiment trends from large volumes of text data, providing insights into public opinion dynamics or customer preferences.

B. *Bidirectional Long Short-Term Memory (Bi-LSTM)*

Bi-LSTM model, derived from the Bidirectional Recurrent Neural Network (Bi-RNN), processes data by analyzing it both forwards and backwards [27]. This method enhances the context understanding of the sequence data. As depicted in Fig. 2, Bi-LSTM uses two separate LSTM layers, one moving forward and the other backward through the input sequence. This dual-pathway ensures comprehensive visibility of data at any point, integrating insights from both before and after the current data point. This extensive perspective highly enhances the model's accuracy and depth of understanding. The Bi-RNN's model employs traditional LSTM gates (forget, input, and output gates) in both directional layers, which allows precise control over information flow.

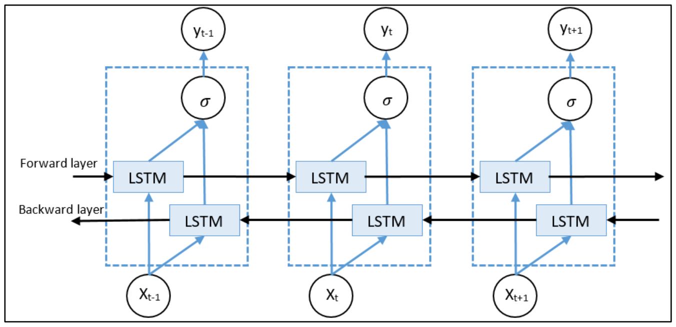


Fig. 2. BiLSTM architectire.

In the context of sentiment analysis using deep learning, Bi-LSTMs are particularly effective due to their ability to understand the full context of expressions, capturing the nuances that influence sentiment. This capability makes them ideal for analyzing extensive text data, such as customer reviews or social media posts, where understanding the sentiment context is key to interpreting overall sentiment accurately.

C. *Gated Recurrent Units (GRU)*

GRUs mark a significant step forward in neural network technology [28]. Like their close relative, the LSTM, GRUs are designed to process sequences of data but with a simplified architecture that includes two key components: the update gate and the reset gate. These two gates are critical to the GRU's function. The update gate determines how much of the past information to keep against the new input, while the reset gate controls the extent to which the previous state affects the current state. This setup allows GRUs to discard irrelevant data, making them efficient and flexible.

GRUs stand out by managing variable-length input sequences, crucial for understanding the nuances in written opinions. Their ability to maintain relevant historical information and combine it with new, incoming data allows for more accurate predictions of sentiment trends. This capability is beneficial in analyzing large volumes of text data, providing deeper insights into consumer sentiments and market trends.

D. *Bidirectional Gated Recurrent Units (Bi-GRU)*

Bi-GRU extends the concept of Bi-RNN by integrating GRU mechanisms for both forward and backward sequence processing [27]. This architecture employs two critical gates: the update gate, which integrates new information, and the reset gate, which controls the amount of past information retained.

In the Bi-GRU setup, the interaction of these gates in both directions allows the model to synthesize information from both past and future contexts relative to the current data point. This approach greatly enhances the model's understanding of sequences, improving its predictive capabilities in applications like sentiment analysis.

E. *Convolutional Neural Network Long Short-Term Memory (CNN-LSTM)*

The CNN-LSTM architecture combines the spatial analysis strengths of Convolutional Neural Networks (CNN) with the sequential data handling capabilities of Long Short-Term Memory (LSTM) networks [29]. The CNN-LSTM model can analyze video or sequential image data. It combines the strengths of CNNs, which capture spatial details from visual data, with LSTMs that track how these features evolve over time. This integrated approach allows for a refined understanding of changes in sentiment. As a result, the CNN-LSTM model is highly effective for analyzing customer reactions in video reviews and social media content, providing nuanced insights into consumer sentiment trends.

F. *Convolutional Long Short-Term Memory (ConvLSTM)*

The ConvLSTM represents an enhancement in neural network architecture by integrating the LSTM's time-sensitive processing capabilities with the spatial feature detection of convolutional layers [29]. This architecture embeds convolutional operations within the LSTM cell transitions, making it particularly adept at managing data that exhibits both spatial and temporal characteristics. For sentiment analysis, particularly in applications like video content analysis, the ConvLSTM excels by capturing temporal sequences of spatial features, such as facial expressions or body language. This ability helps in accurately determining the progression of emotions or sentiments over time.

IV. MODEL DEVELOPMENT

This section describes our proposed methods for sentiment analysis. It outlines the overall framework of deep learning, encompassing all stages from data collection to model optimization.

A. *Deep Learning Framework for Sentiment Analysis*

Our proposed system architecture is structured into four phases, as depicted in Fig. 3. This architecture is crafted to process and interpret sentiments efficiently. The initial phase encompasses data collection and preprocessing, which includes text cleaning, stop word removal, and lemmatization. These steps aim to enhance the data's quality and relevance. Subsequently, the focus shifts to the critical task of feature extraction, employing sophisticated techniques such as TF-IDF, Word2Vec, and GloVe to identify meaningful patterns in the data. In the third phase, we concentrate on developing and rigorously training a variety of machine learning models. The fourth and concluding phase involves a comparative evaluation of the models' performances. This comparison is vital for determining the most effective methods in terms of accuracy and efficiency, thereby identifying the best strategy for sentiment analysis. The details of these phases are elaborated in the following subsections.

B. *Dataset Description*

Our research utilizes the Sentiment140 dataset, a significant contribution from Stanford University [16]. Known for its comprehensive and carefully assembled collection of tweets, the dataset is gathered directly from Twitter through its search API. It stands out for its utility in sentiment analysis research and is publicly accessible on Kaggle. Kaggle is a platform renowned for hosting a wide array of datasets suitable for various data science projects. The Sentiment140 dataset is especially valuable for training machine learning models in sentiment analysis, thanks to its large size and balanced composition. It features 1.6 million tweets, evenly split between positive and negative sentiments.

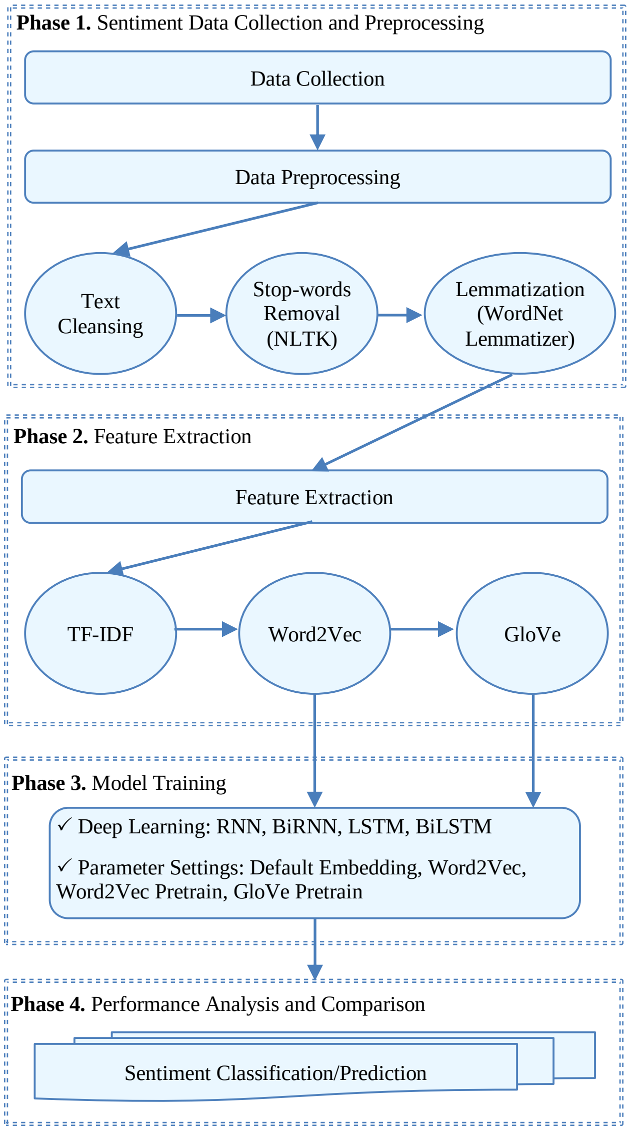
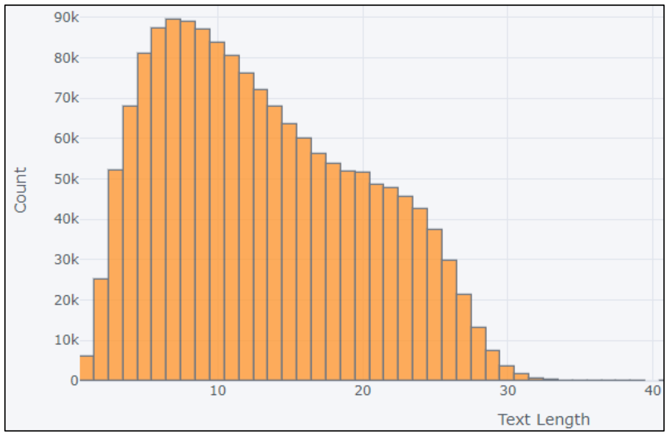

Phase 1. Sentiment Data Collection and Preprocessing
Data Collection
Data Preprocessing
Text Cleansing
Stop-words Removal (NLTK)
Lemmatization (WordNet Lemmatizer)
Phase 2. Feature Extraction
Feature Extraction
TF-IDF
Word2Vec
GloVe
Phase 3. Model Training
✓ Deep Learning: RNN, BiRNN, LSTM, BiLSTM
✓ Parameter Settings: Default Embedding, Word2Vec, Word2Vec Pretrain, GloVe Pretrain
Phase 4. Performance Analysis and Comparison
Sentiment Classification/Prediction

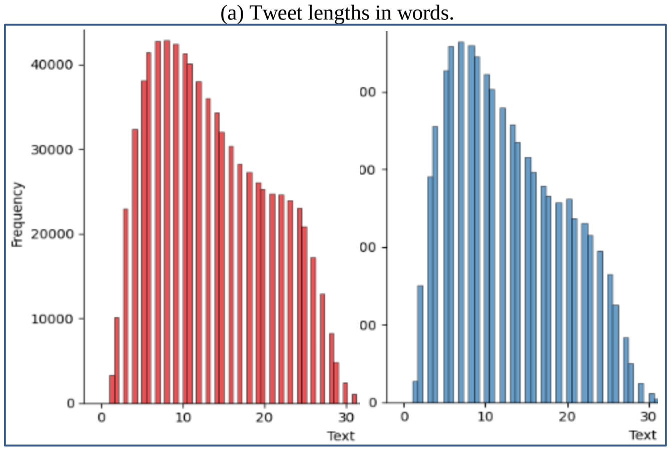
Fig. 3. Deep learning framework for sentiment analysis.

The structure of this dataset is meticulously organized into a CSV file format, which includes six critical columns: Sentiment, Id, Date, Query, User, and OriginalTweet. The 'Sentiment' column classifies each tweet's emotional tone with a numeric system: '0' denotes negative sentiment, and '4' represents positive sentiment. The 'Id' column provides a unique identifier for each tweet. The 'Date' column records the tweet's posting date. The 'Query' column specifies if the tweet was retrieved using a specific search keyword, though our study includes all tweets regardless of the query used. The 'User' column lists the username of the tweet's author, and 'OriginalTweet' contains the text of the tweet. For our analysis, we focus solely on the 'Sentiment' and 'OriginalTweet' columns. This selective approach allows us to concentrate on the textual content and its associated sentiment, discarding extraneous data that do not directly contribute to our sentiment analysis objectives.

Through statistical analysis of the dataset, we display the frequency distribution of tweet lengths in Fig. 4 and Fig. 5. Fig. 4 unveils the range of tweet lengths, highlighting the concise nature of Twitter communication. Most tweets are brief, peaking at seven words. This pattern highlights the importance of grasping the typical tweet structure and tailoring our analysis techniques to Twitter's compact format. Fig. 5 reveals the dataset's lexical patterns, offering insights into the vocabulary frequently used by Twitter users. This analysis is crucial for pinpointing key terms commonly found in tweets, guiding our preprocessing and feature extraction strategies to improve model performance.



(a) Tweet lengths in words.



(b) Length of negative (left) and positive (right) tweets.

Fig. 4. Frequency distribution of tweet lengths.

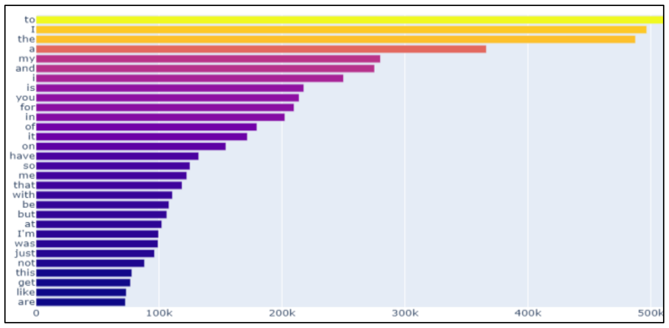


Fig. 5. Frequency distribution of the 30 most common words.

C. Data Preprocessing

Data preprocessing is a critical step in our sentiment analysis methodology, addressing the challenges posed by the unstructured or semi-structured nature of data harvested from online platforms like Twitter. Prior research [17] has shown that effective preprocessing of data plays a key role in enhancing the accuracy of machine learning (ML) models. It does so by eliminating noise and reducing the dataset's dimensionality, which brings into focus the features that have a high correlation with the target outcomes. Moreover, given the computational demands of processing large datasets, preprocessing not only aids in improving prediction accuracy by concentrating on relevant data but also enhances computational efficiency [18]. The details of this data preprocessing phase are outlined in the following subsections.

1) Data cleansing: In the first step of data cleansing, we utilize Python's "re" module for its robust regular expressions. These expressions enable us to methodically eliminate URLs, HTML tags, hashtags, mentions, emojis, and unnecessary spaces from the dataset. This crucial step helps in eliminating distractions and standardizing the data for analysis. Next, we remove special characters and numbers since they typically don't aid sentiment analysis, further purifying the dataset. We also standardize all words to lowercase to avoid duplicates that could diminish the performance, for example, treating capitalized words at the start of sentences the same as their lowercase counterparts elsewhere. This approach ensures uniform treatment of words, regardless of their position in a sentence.

Given Twitter's informal and abbreviated language, we employ a detailed list of abbreviations to translate shortened forms into their full expressions, such as converting "he's" to "he is". This standardization is crucial for maintaining data consistency and clarity. Moreover, we rectify spelling errors resulting from repeated characters, for example, correcting “saddd” to “sad”. For a more straightforward classification process, we adjust sentiment labels, designating negative sentiments as '0' and positive sentiments as '1'. Table I illustrates the distribution of tweet lengths after cleansing, laying the groundwork for further analysis.

TABLE I. TWEET LENGTH DISTRIBUTION AFTER CLEANSING

Statistic	Original Tweet	Cleansed Tweet
The average value	13.18	11.69
Standard deviation	6.96	6.46
Minimum length	1	0
25%	7	6
50%	12	11
75%	19	17
Maximum length	64	40

2) Stop word removing: Stop words like “is”, “has”, “and”, “to”, and others frequently appear in sentences and may reduce the significance of other words in sentiment analysis. Removing these stop words is a common strategy to decrease noise in text data. However, in sentiment analysis, this practice might change the intended meaning of sentences. For example, “The product is not good” clearly expresses a negative sentiment, but removing the stop word changes it to “product good”, suggesting a positive sentiment instead. To assess how stop word removal affects model performance, we explore two scenarios in our study: one with stop word removal and one without.

For this procedure, we utilize the stop word dictionary from the Natural Language Toolkit library, available at www.nltk.org, with a crucial modification: we retain the words “not” and “no” to preserve the sentiment context within the sentences. This method ensures that tweets are cleansed of stop words while preserving essential words for expressing negation. Tables II and III provide insights into the impact of this step. Table II lists the top 10 most frequent words in the dataset before cleansing, highlighting that stop words dominate the list across both negative and positive sentiments with similar frequencies. Table III then illustrates how tweet lengths change once stop words are removed, offering a quantitative view of this preprocessing step's effect.

TABLE II. TOP 10 MOST FREQUENT WORDS IN ORIGINAL TWEETS

Word	Frequency in Negative Tweets	Frequency in Positive Tweets
to	613,036	492,288
the	482,000	493,002
a	351,648	380,776
my	333,834	226,216
i	320,264	179,768
and	280,480	270,046
is	236,252	199,134
in	216,842	187,746
for	192,596	227,006
it	182,174	161,450

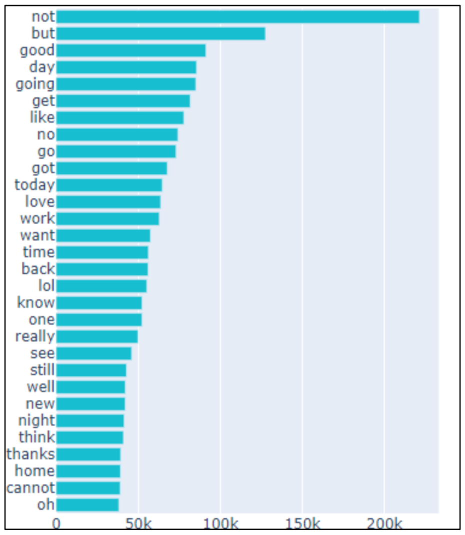
TABLE III. TWEET LENGTH DISTRIBUTION AFTER CLEANSING AND STOP WORD REMOVAL

Statistic	Cleansed Tweets	Tweets without stop words
The average value	11.69	7.07
Standard deviation	6.46	3.89
Minimum length	0	0
25%	6	4
50%	11	7
75%	17	10
Maximum length	40	34

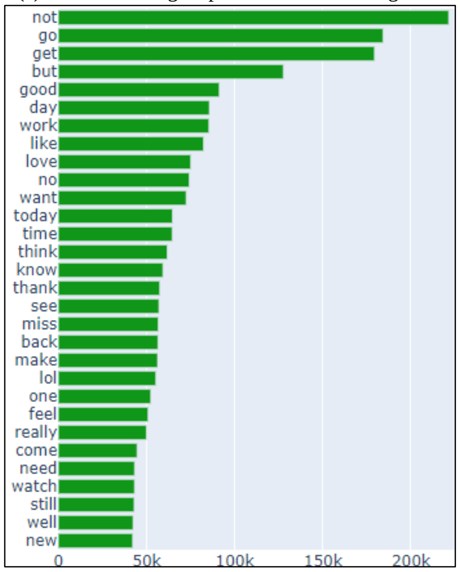
3) Word normalization: English words frequently appear in multiple forms. For example, “go”, “went”, “gone”, “going”, and “goes” all stem from “to go”. If these variations are not simplified, they can unnecessarily expand the dataset with redundant features. To address this, we use the NLTK library

[19] to reduce words to their base forms, employing two approaches: stemming and lemmatization.

Stemming shortens words by removing endings or beginnings, which may sometimes lead to imprecise meanings or spellings. This method is preferred in large datasets where processing speed is crucial. Conversely, lemmatization considers the word's context to derive its meaningful base form, known as the lemma. Although more accurate, lemmatization requires more computational resources because it involves extensive lookup tables. Fig. 6 demonstrates how word normalization simplifies "going" to its fundamental form "go". This crucial step prepares the dataset for the next phase of feature extraction, making the text more concise.



(a) After removing stop words and cleansing text.



(b) After Lemmatization.

Fig. 6. Top 30 words after normalization to root forms.

D. Feature Extraction

Following the initial preprocessing phase, our dataset undergoes feature extraction, a pivotal step in transforming text into a format amenable for model training. We employ three advanced techniques for this purpose: TF-IDF, Word2Vec, and GloVe, each converting text into numerical vectors.

1) TF-IDF: The Term Frequency-Inverse Document Frequency (TF-IDF) stands out for its ability to identify the significance of a word within a document, relative to a collection of documents. It calculates a weight for each term: the Term Frequency (TF) measures a term's frequency within a document, while the Inverse Document Frequency (IDF) assesses the term's rarity across all documents. The formula given as Eq. (1) combines these two metrics to determine a term's overall importance:

$$\mathrm{TF}-\mathrm{ID}F_{(t,d)} = \mathrm{T}F_{(t,d)} \times \log\left(\frac{N}{DF_{(t)}}\right). \quad (1)$$

Here, N represents the total document count in the dataset, $DF_{(t)}$ denotes the number of documents featuring term t, and $TF_{(t,d)}$ is term t's frequency in document d. Utilizing the Scikit-learn library's TF-IDF vectorizer [21], available at scikit-learn.org, we efficiently extract features that prioritize words based on their document-wise relevance, reducing emphasis on common words and elevating unique terms. Configurations such as *min_df=5* exclude terms appearing in fewer than five documents, and *ngram_range=(1,1)* limits our focus to individual words.

2) Word2Vec: Word2Vec, a model for creating word embeddings from text, uses neural networks in two distinct approaches: Continuous Bag of Words (CBOW) and Skip-gram [22]. The CBOW method predicts a word based on its surrounding context, while the Skip-gram approach does the opposite by predicting the surrounding context of a word. These methods not only make the model more versatile but also enhance its understanding of language subtleties.

In this research, we train Word2Vec on our dataset with the help of the Gensim library [21]. This process generates dense and meaningful vector representations of words. Additionally, we utilize pre-trained vectors from Google News. This dataset contains about 100 billion words, which have been used to produce 300-dimensional vectors for over 3 million terms. Such extensive data enrich our analysis by providing a wide range of linguistic insights.

Both the Gensim library and the Google News vectors are accessible through resources like the Gensim library itself and the Kaggle platform [21]. These tools and datasets play a crucial role in our methodology. Table IV provides detailed information on our setup and parameters.

TABLE IV. WORD2VEC TRAINING SETUP PARAMETERS

Parameter	Value	Description
vector_size	300	Dimension of word vector
workers	8	Number of threads involved in model training
min_count	5	Exclude words appearing fewer than 5 times
sg	0	Use CBOW

3) GloVe: GloVe, short for Global Vectors, emerges as an influential open-source initiative from Stanford, as noted in

[23]. This innovative project offers a method for generating word vector representations, facilitating a deeper understanding of language through mathematical modeling. Unlike traditional models, GloVe constructs word embeddings by optimizing a model based on the aggregation of word co-occurrences across a text corpus. This method focuses on shrinking the dimensions of the occurrence count matrix, capturing the essence of word relationships more efficiently.

Our research benefits from the utilization of GloVe vectors pre-trained on the extensive Common Crawl dataset. This massive corpus, comprising 840 billion tokens and a vocabulary of 2.2 million terms, provides a rich, contextually diverse linguistic foundation. The downloaded dataset, encapsulating 300-dimensional vectors for words and phrases, spans 2.03 GB, offering a comprehensive resource for our analytical needs.

E. Deep Learning Configurations and Parameter Tuning

Among the six deep learning architectures discussed in Section III, BiLSTM stands out for its performance in sentiment analysis [24]. For this reason, and to facilitate direct comparisons with previous research, we focus our in-depth experiments on BiLSTM. The key advantage of BiLSTM is its bidirectional data processing capability, which allows it to effectively assimilate contextual information from both preceding and subsequent text segments. This dual-directional approach is particularly beneficial for sentiment analysis, where understanding the complete context of text sequences is crucial for accurately determining sentiment polarity.

Building on this foundation, we have implemented three distinct BiLSTM models, each configured with different layer setups to optimize performance based on the nature of sentiment-laden words within the text, as noted in [24]. These configurations are specifically designed to enhance the model's ability to detect and interpret sentiment polarity, which heavily relies on contextual cues. The detailed specifications of each model configuration are outlined in Table V, showing the variations in layer structures and their intended impacts on model efficacy.

TABLE V. CONFIGURATIONS OF BILSTM MODELS

Model	Parameters and architecture
BiLSTM1	Embedding layer, Bidirectional LSTM x 2, Conv1D, GlobMaxPool1D, Dense(16, ReLU), Dense(2, softmax)
BiLSTM2	Embedding layer, Conv1D, Maxpooling1D, BiRdirectional LSTM, Dropout, Dense(2, softmax)
BiLSTM3	Embedding layer, Bidirectional LSTM, Dense (128, ReLU), Dropout, Dense (64, ReLU), Dense (2, softmax)

To ensure optimal performance of the BiLSTM deep learning algorithm, we utilize the GridSearchCV tool from the Sklearn library for meticulous parameter fine-tuning. This process involves 10-fold cross-validation solely on the training set to rigorously evaluate different configurations without risking leakage from the testing data. Through this approach, we have identified and implemented a set of optimal parameters that significantly enhance model efficacy. These parameters include a learning rate of 0.001, a training duration of 50 epochs, and a batch size of 1024, using the Adam optimizer for efficient convergence. Additionally, to prevent overfitting, we incorporate an EarlyStopping mechanism with a patience of 5 epochs, halting training if there is no improvement in the validation loss. Furthermore, we deploy the ReduceLROn-Plateau strategy, which automatically reduces the learning rate when there are no further improvements in validation loss, ensuring that the training process is both efficient and robust.

V. EXPERIMENTAL ANALYSIS AND COMPARISON

This section conducts a detailed exploration of experimental tasks, emphasizing the practical use of the methods described in Section III. We train various deep learning models, each employing various configurations and parameters meticulously optimized for sentiment classification. We evaluate the effectiveness of these models using recognized evaluation metrics, including Accuracy, Precision, Recall, and F1-Scores.

The results from these experiments lay the groundwork for in-depth analysis, discussion, and comparison. By delving into these outcomes, we aim to identify the strengths and weaknesses of each model configuration and evaluate their influence on overall performance. This analysis is crucial as it pinpoints the most effective techniques and settings tailored to the unique characteristics of our selected dataset. Moreover, our research extends beyond basic performance metrics to incorporate a comparative analysis of the models. We contrast the models against one another under equivalent conditions to determine which configurations deliver the optimal balance between precision and recall and which enhance overall accuracy and F1-Scores. This comprehensive experimental analysis also aims to establish benchmarks for sentiment classification, which is detailed in the subsequent subsections.

A. Analysis of Traning Performance

In our evaluation of deep learning models, we repeatedly train and validate each model ten times to compute both the average values and standard deviations. As detailed in Table VI, the BiLSTM2 model demonstrates superior performance, achieving an accuracy of 88.881% and an AUC of 95.996%, which are the highest among the tested BiLSTM models. In contrast, the BiLSTM2 Word2Vec Pretrain model shows the lowest performance, with an accuracy of 81.351% and an AUC of 89.515%.

TABLE VI. PERFORMANCE OF BILSTM MODELS ON THE TRAINING DATA

Model	Accuracy	AUC	Loss
BiLSTM1	0.85228 ±0.00226	0.93258 ±0.00197	0.32977 ±0.00482
BiLSTM2	0.88881 ±0.00077	0.95996 ±0.00043	0.25562 ±0.00141
BiLSTM3	0.84987 ±0.00022	0.93081 ±0.00021	0.33406 ±0.00052
BiLSTM1_Word2Vec	0.83079 ±0.00120	0.91170 ±0.00100	0.37321 ±0.00208
BiLSTM2_Word2Vec	0.81789 ±0.00208	0.89966 ±0.00209	0.39787 ±0.00416
BiLSTM3_Word2Vec	0.83011 ±0.00381	0.91252 ±0.00331	0.37429 ±0.00671
BiLSTM1_Word2Vec_Pretrain	0.82740 ±0.00128	0.90981 ±0.00118	0.37989 ±0.00237
BiLSTM2_Word2Vec_Pretrain	0.81351 ±0.00112	0.89515 ±0.00112	0.40666 ±0.00208
BiLSTM3_Word2Vec_Pretrain	0.82541 ±0.00317	0.90799 ±0.00299	0.38340 ±0.00601

BiLSTM1_Glove_Pretrain	0.83309 ±0.00143	0.91501 ±0.00012	0.36939 ±0.00262
BiLSTM2_Glove_Pretrain	0.81906 ±0.00081	0.90035 ±0.00088	0.39682 ±0.00170
BiLSTM3_Glove_Pretrain	0.83018 ±0.00565	0.91119 ±0.00516	0.37443 ±0.01084

B. *Analysis of Testing Performance*

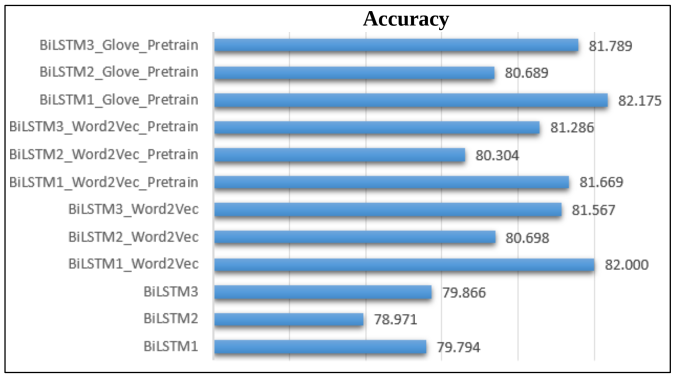
Fig. 7 illustrates the performance metrics of various BiLSTM models using Word2Vec and GloVe embeddings. The BiLSTM1 model with GloVe embeddings shows the best performance, achieving an accuracy of 82.175% (see Fig. 7(a)), an F1-Score of 82.174% (see Fig. 7(b)), a precision of 82.189% (see Fig. 7(c)), and a recall of 82.178% (see Fig. 7(d)). In contrast, the BiLSTM2 model with a default embedding layer records the lowest metrics: an accuracy of 78.971% (see Fig. 7(a)), an F1-Score of 78.929% (see Fig. 7(b)), a precision of 78.926% (see Fig. 7(c)), and a recall of 78.918% (see Fig. 7(d)). Overall, Fig. 7 highlights the superior performance of the BiLSTM1 model with GloVe embeddings across all measured metrics.

1) Performance with SGD and adam optimizers: To enhance the BiLSTM1 model, we explore variations such as BiLSTM1, BiLSTM1_Word2Vec, BiLSTM1_Word2Vec_ Pre-train, and BiLSTM1_Glove_Pretrain using the SGD optimizer with settings of 50 epochs, a learning rate of 0.1, momentum of 0.8, and Nesterov disabled. Subsequently, we evaluate these models against their counterparts trained with the Adam optimizer. The detailed outcomes of these experiments are presented in Fig. 8 and Fig. 9.

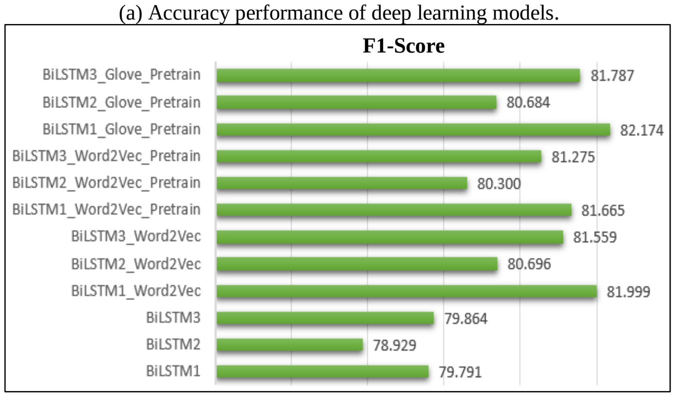
As depicted in these figures, the standard BiLSTM1 model trained with the SGD optimizer has lower accuracy and F1-Score than with Adam. Specifically, Adam achieves 79.794% accuracy, surpassing SGD's 79.198%. Similarly, the BiLSTM1_Word2Vec model shows better performance with Adam, reaching an accuracy of 82% and an F1-Score of 81.999%.

Further analysis shows the BiLSTM1_Word2Vec_Pretrain model, using pre-trained Word2Vec vectors, performs similarly to its non-pretrained counterpart. On the other hand, the BiLSTM1_Glove_Pretrain model, with pre-trained GloVe embeddings, outperforms all others, achieving the highest accuracy of 82.175% and an F1-Score of 82.174%.

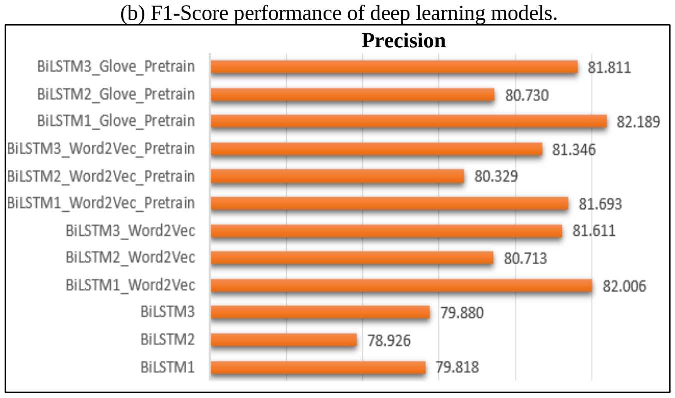
These findings underscore the advantage of using pre-trained embeddings like Word2Vec and GloVe. Additionally, the Adam optimizer tends to yield superior results compared to SGD, highlighting its effectiveness in optimizing deep learning models.



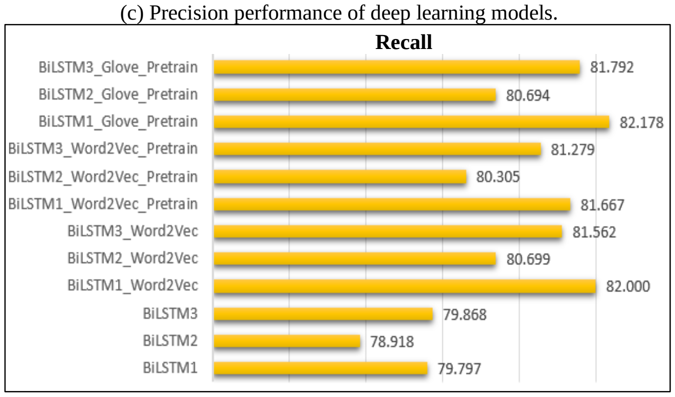
(a) Accuracy performance of deep learning models.



(b) F1-Score performance of deep learning models.



(c) Precision performance of deep learning models.



(d) Recall performance of deep learning models.

Fig. 7. Performance of deep learning models on the testing data.

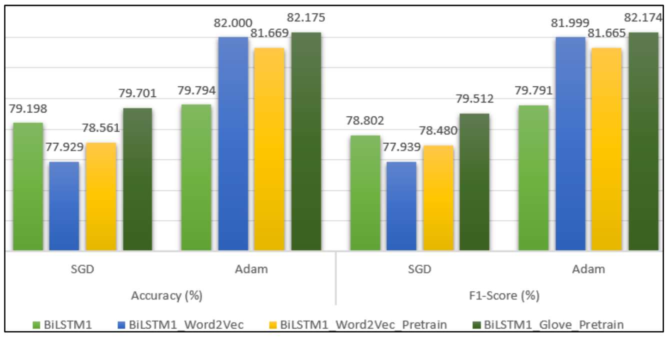


Fig. 8. BiLSTM1 performance with SGD optimizer.

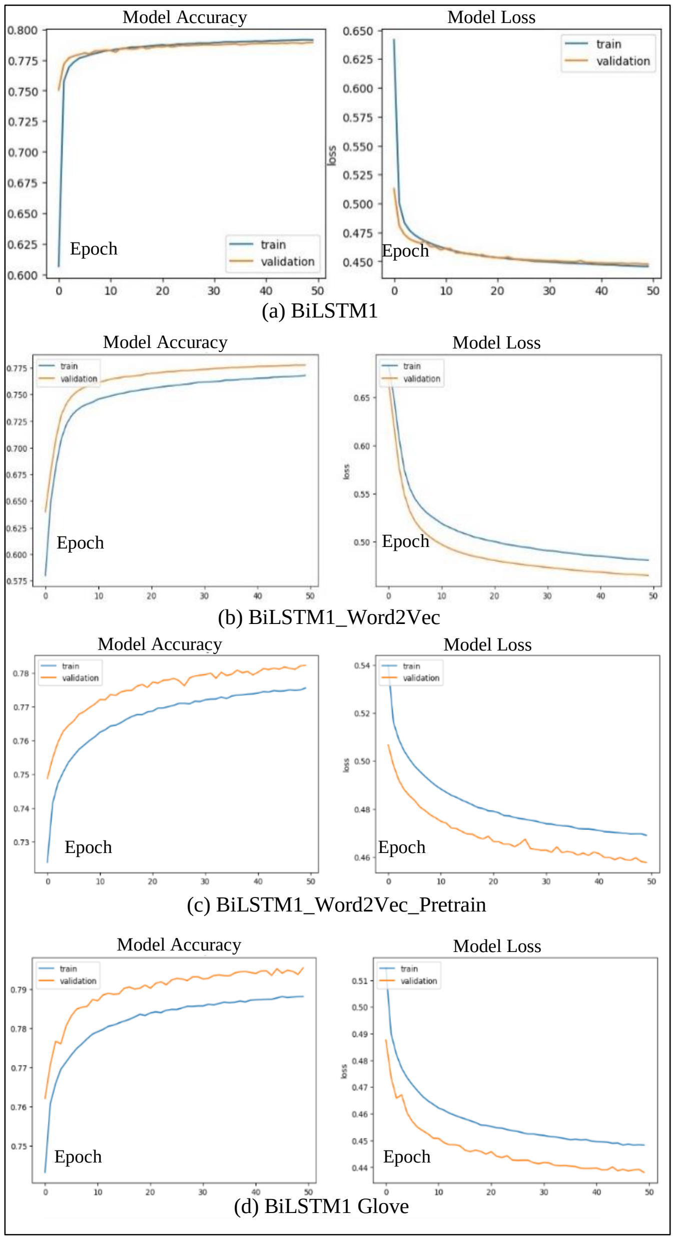


Fig. 9. BiLSTM1 accuracy trends across epochs with SGD optimizer.

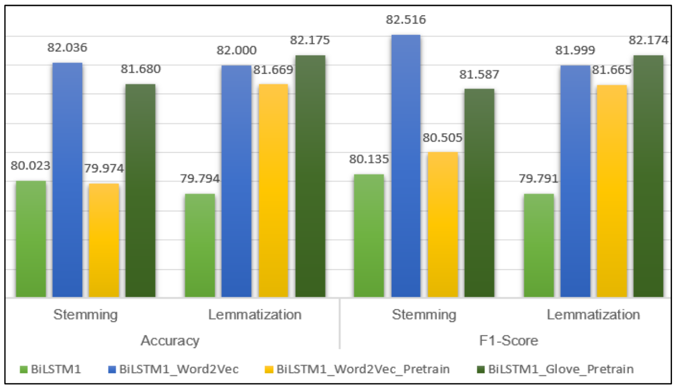


Fig. 10. BiLSTM1 performance with Stemming and Lemmatization.

2) Performance with stemming and lemmatization: We evaluate how Stemming and Lemmatization impact the performance of various BiLSTM1 configurations: BiLSTM1, BiLSTM1_Word2Vec, BiLSTM1_Word2Vec_Pre-train, and BiLSTM1_Glove_Pretrain. We analyze and compare these preprocessing techniques to determine which yields better results, with specifics illustrated in Fig. 10.

For the standard BiLSTM1 model, both Stemming and Lemmatization have negligible effects on performance, achieving similar accuracy and F1 scores: The method achieves an accuracy of 80.023% and an F1 score of 80.135%. The BiLSTM1_Word2-Vec, incorporating Word2Vec, also shows little variation between the two techniques, with a minor deviation of just 0.0036%.

Similarly, the BiLSTM1_Word2Vec_Pretrain and BiLSTM1 models exhibit minimal differences when applying either tech-nique. However, Lemmatization provides a slight improvement in performance, achieving an accuracy of 81.669% and an F1-score of 81.665%.

The BiLSTM1_Glove_Pretrain model, using pre-trained GloVe embeddings, performs well under both techniques but shows a slight preference for Lemmatization, which delivers the highest accuracy and F1-score among the tested models at 82.175% and 82.174%, respectively.

The comprehensive analysis indicates that although the differences between the two methods are generally small across the models, Lemmatization consistently shows a slight improvement in accuracy and F1-scores.

3) Performance with stop words: In this experiment, we investigate how the exclusion of stop words influences the performance of the BiLSTM1 model, particularly focusing on the BiLSTM1_Glove_Pretrain model, which omits the stop word removal step during data preprocessing. The results from this configuration demonstrate an accuracy of 0.83962, an F1-Score of 0.83857, a recall of 0.83042, and a precision of 0.84689. These findings suggest that removing stop words can significantly affect model performance in sentiment analysis tasks, particularly with techniques that rely heavily on word context, like Word2Vec.

Notably, the performance metrics for the BiLSTM1_Glove_ Pre-train model show substantial improvement across all parameters when stop words are retained: accuracy improved from 0.82175 to 0.83962, F1-Score from 0.82174 to 0.83857, recall from 0.82178 to 0.83042, and precision from 0.82189 to 0.84689. This improvement highlights how stop words add contextual depth, enhancing the model's accuracy.

4) Benchmarking against prior studies: Our study meticulously compares the effectiveness of our sentiment analysis models with the results reported in a previous study, specifically [25], which employed the same dataset and data division methodology. The dataset is partitioned into training and testing sets with a 90:10 ratio, and the training set is further split into training and validation sets, also with a 90:10 ratio.

The results in Table VII demonstrate that our proposed methods (BiLSTM1_Glove_Pretrain and BiLSTM1_Glove_ Pretrain With-out Stop Word Removal), shown in the first two rows, consistently outperform the approaches from study [25] (listed in the subsequent rows) in terms of accuracy and F1-score. Notably, our methods achieve, on average, an improvement of 2.07% in accuracy and 2.20% in F1-score compared to those reported in study [25].

To elucidate, our BiLSTM1_Glove_Pretrain model records an accuracy of 82.2% and an F1-score of 82.2%, while our BiLSTM1_Glove_Pretrain_NoSW-Removal variant shows even more impressive results with an accuracy of 83.9% and an F1-score of 83.8%. In contrast, the best-performing model from the prior study, the LSTM + FastText, only achieves an accuracy and F1-score of 82.4%. Other models from the same study, such as LSTM + Glove and LSTM + Glove Twitter, present lower performances with accuracy and F1-scores ranging from 80.4% to 81.6%. These results underscore the effectiveness of our methodologies, particularly in enhancing the precision and reliability of sentiment analysis in real-world applications.

TABLE VII. PERFORMANCE COMPARISON WITH PRIOR STUDY [25]

Model	Accuracy	F1-score
BiLSTM1_Glove_Pretrain _	82.2 %	82.2 %
BiLSTM1_Glove_Pretrain_NoSW-Removal	**83.9 %**	**83.8 %**
DNN (Baseline) [25]	79.0 %	78.4 %
LSTM + FastText [25]	**82.4 %**	**82.4 %**
LSTM + Glove [25]	81.5 %	81.4 %
LSTM + Glove Twitter [25]	80.4 %	80.4 %
LSTM + w/o Pretrained Embed [25]	81.6 %	81.4 %

VI. CONCLUSION AND FUTURE DIRECTIONS

Social networks such as Twitter, now known as *X*, are crucial platforms for capturing real-time public sentiments. This study exploited the power of these platforms, particularly utilizing the Sentiment140 dataset, which includes 1.6 million tweets, to develop and evaluate a comprehensive methodology for sentiment analysis using advanced machine learning techniques. Our approach spanned from data collection and preprocessing to feature extraction and model optimization. We extensively explored several deep learning architectures through various configurations and parameters settings.

Our exploration into deep learning frameworks, particularly the BiLSTM models, revealed their high ability to capture nuanced expressions of sentiment. These models, when integrated with pre-trained GloVe embeddings, significantly outperformed traditional embeddings, achieving an accuracy of 88.88% and an AUC of 96%. These results highlight the potential of deep learning techniques to enhance sentiment analysis tools.

The evaluations not only confirmed the effectiveness of our methodology but also helped establish benchmarks in the field. Compared to existing approaches, our methods consistently demonstrated higher performance, often surpassing baseline results by more than 3%. This provides valuable insights and a solid foundation for further research and practical applications.

Our research will increasingly focus on exploring deep learning techniques, particularly Transformer-based models, which are well-suited for managing the complexities of language in sentiment analysis due to their superior handling of sequential data. We also aim to expand our methodologies to include multilingual datasets, enhancing the global applicability of our findings across various linguistic and cultural contexts. These strategic directions are intended to not only advance the technical aspects of sentiment analysis but also to increase its practical relevance and effectiveness in dynamic environments.

ACKNOWLEDGMENT

This work has been supported by the College of Information Technology and Communication at Can Tho University. Additionally, we received support from the European Union's Horizon Research and Innovation program under the MSCA-SE grant agreement 101086252, Call: HORIZON-MSCA-2021-SE-01.

REFERENCES

[1] U. Singh, K. Abhishek, and H.K. Azad. 2024, “A Survey of Cutting-edge Multimodal Sentiment Analysis”, *ACM Comput. Surv*, 2024.

[2] D. Dash, M. Kolekar, C. Chakraborty, and R. Khosravi, “Review of Machine and Deep Learning Techniques in Epileptic Seizure Detection using Physiological Signals and Sentiment Analysis”. *ACM Trans.* 23, 1, Article 16, 2024.

[3] R. Das and T.D. Singh, “Multimodal sentiment analysis: A survey of methods, trends, and challenges”, *ACM Comput.* 55, 13s, 2023.

[4] M. Ibánez, A. Ventura, F. Mateos, P. Jiménez, “A review on sentiment analysis from social media platforms”, *Expert Systems with Applications*, Vol. 223, 2023.

[5] Bordoloi and Biswas, “Sentiment analysis: A survey on design framework, applications and future scopes”, *Artif Intell Rev 56*, 2023.

[6] Y. Li *et al.*, “TransRegex: multi-modal regular expression synthesis by generate-and-repair”, in *International Conference on Software Engineering (ICSE)*, IEEE, pp. 1210–1222, 2021.

[7] D. J. Ladani and N. P. Desai, “Stopword identification and removal techniques on TC and IR applications: A survey,” in *International Conference on Advanced Computing and Communication Systems (ICACCS)*, IEEE, pp. 466–472, 2020.

[8] N. Zukarnain, B. S. Abbas, S. Wayan, A. Trisetyarso, and C. H. Kang, “Spelling checker algorithm methods for many languages”, in *International Conference on Information Management & Technology (ICIMTech)*, IEEE, pp. 198–201, 2019.

[9] Md. Kowsher, A. Tahabilder, M. M. Hossain Sarker, Md. Z. Islam Sanjid, and N. J. Prottasha, "Lemmatization algorithm development for bangla natural language processing", in *icIVPR*, IEEE, pp. 1–8, 2020.

[10] J. S. Santos, A. Paes, and F. Bernardini, "Combining labelled datasets for sentiment analysis from different domains based on dataset similarity to predict electors sentiment" in *Brazilian Conference on Intelligent Systems (BRACIS)*, IEEE, pp. 455–460, 2019.

[11] E. Haddi, X. Liu, and Y. Shi, "The role of text pre-processing in sentiment analysis", *Procedia Comput. Sci.*, vol. 17, pp. 26–32, 2013.

[12] K. K. Kiilu, G. Okeyo, R. Rimiru, and K. Ogada, "Using Naïve Bayes algorithm in detection of hate Tweets," *Int. J. Sci. Res. Publ. IJSRP*, vol. 8, no. 3, 2018.

[13] W. P. Ramadhan, S. Astri Novianty, and S. Setianingsih, "Sentiment analysis using multinomial logistic regression," in *ICCREC*, IEEE, pp. 46–49, 2017.

[14] U. Gandhi, P. Kumar, G. Babu, and G. Karthick, "Sentiment analysis on Twitter data by using convolutional neural network and long short-term memory (LSTM)", *Wirel. Pers. Commun.*, 2021.

[15] V. Umarani, A. Julian, and J. Deepa, "Sentiment analysis using various machine learning and deep learning techniques", *J. Niger. Soc. Phys. Sci.*, pp. 385–394, Nov. 2021.

[16] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision", *CS224N Proj. Rep. Stanf.*, vol. 1, 2019.

[17] F. Rustam, I. Ashraf, A. Mehmood, S. Ullah, and G. S. Choi, "Tweets classification on the base of sentiments for US airline companies", *Entropy*, vol. 21, no. 11, p. 1078, 2019.

[18] V. Kalra and R. Aggarwal, "Importance of text data preprocessing & implementation in RapidMiner", *ICITKM*, vol. 14, pp. 71–75, 2017.

[19] D. Khyani and S. B. S, "An interpretation of lemmatization and stemming in natural language processing", *J. Univ. Shanghai Sci. Technol.*, 2020.

[20] S. Robertson, "Understanding inverse document frequency: on theoretical arguments for IDF", *J. Doc.*, vol. 60, no. 5, 2004.

[21] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python", *ArXiv12010490 Cs*, 2021.

[22] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space", *arXiv*, 1301.3781, 2013.

[23] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation", in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

[24] G. Xu, Y. Meng, X. Qiu, Z. Yu, and X. Wu, "Sentiment analysis of comment texts based on BiLSTM", *IEEE Access*, vol. 7, 2019.

[25] A. S. Imran, S. M. Daudpota, Z. Kastrati, and R. Batra, "Cross-cultural polarity and emotion detection using sentiment analysis and deep learning on COVID-19 related Tweets", *IEEE Access*, vol. 8, pp. 181074–181090, 2020.

[26] U.B., Mahadevaswamy and P. Swathi, "Sentiment analysis using bidirectional LSTM network". *Procedia Computer Science*, 45-56, 2023.

[27] R. Cheruku, K. Hussain, I. Kavati, A.M. Reddy, and K.S. Reddy, "Sentiment classification with modified RoBERTa and recurrent neural networks", *Multimedia Tools and Applications*, 83(10), 2024.

[28] K. Cho, B. Merri¨enboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, "Learning phrase representations using rnn encoderdecoder for statistical machine translation", *arXiv, 1406.1078*, 2014.

[29] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S.Corrado, A. Davis, J. Dean, M. Devin, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems", *arXiv, preprint arXiv:1603.04467*, 2016.