Decoding Emotions in Tweets: Deep Learning Approach for Sentiment Analysis

Sharmin Sultana
Dept. Computer Science
University of Massachusetts Lowell
Sharmin_Sultana@student.uml.edu

***Abstract*—Deep Learning, a subset of machine learning in Natural Language Processing (NLP), has significantly elevated the capabilities of sentiment analysis. Utilizing advanced neural network architectures, deep learning facilitates a more accurate and nuanced interpretation of emotions in text. This marks a substantial improvement over traditional NLP methods, which often falter in capturing the intricacies of human language. Our study focuses on a comprehensive sentiment analysis of a large-scale dataset comprising 1.6 million Twitter posts. We employ a range of deep learning models, including CNN, Simple RNN, LSTM, Bi-directional LSTM, and notably, the BERT transformer model. Our primary goal is to classify tweets into positive or negative sentiments, rigorously comparing the effectiveness and accuracy of traditional deep learning approaches against the advanced capabilities of BERT. In our evaluation, models like CNN and LSTM show commendable performance with 82% and 83% accuracy, respectively. However, it's the BERT model that stands out, surpassing other models with a significant performance boost of approximately $\sim$4%. This project not only demonstrates the robustness of deep learning in interpreting complex language patterns but also underscores the transformative impact of models like BERT in the realm of sentiment analysis.**

I. Introduction

Sentiment Analysis (SA), also known as emotion AI or opinion mining, is an intricate blend of text analysis, Natural Language Processing (NLP), and statistical methods aimed at deciphering the sentiments behind user-generated content [1]. It effectively discerns the underlying feelings, thoughts, or attitudes expressed about entities like individuals, situations, or products. This NLP technique is pivotal for identifying whether data or information conveys positive, neutral, or negative sentiments. Businesses and industry experts commonly utilize SA to monitor brand reputation and understand customer needs [2], [3], leveraging the vast amounts of information generated online by an ever-growing number of internet users [4], [5].

In the realm of social media, platforms such as Twitter, Instagram, Facebook, LinkedIn, and YouTube have become global stages for people to express their opinions [6], [7]. Twitter, particularly, stands out as a preferred micro-blogging platform, where users share their views in concise messages known as tweets [8]. With its substantial user base – 152 million daily and 330 million monthly active users – Twitter generates a significant volume of sentiment data through over 500 million daily tweets. These tweets not only offer a rich resource for sentiment analysis but also significantly influence various aspects of life, making Twitter an impactful online social network (OSN) for information dissemination and user interaction [9].

The challenge in analyzing such a vast quantity of data has been traditionally addressed by NLP techniques, which enable the extraction of meaningful information from tweets. Traditional methods like SVM (Support Vector Machine), MNB (Multinomial Naïve Bayes), LR (Logistic Regression), and NB (Naïve Bayes) have shown promise but are often slow and labor-intensive [10], [11].

To address these limitations, Deep Learning (DL), a subset of Machine Learning (ML), has been introduced for more effective classification of Twitter sentiments. DL, which relies on complex algorithms and minimal human intervention, processes vast datasets with increased efficiency and accuracy. This advancement has opened new avenues in sentiment analysis applications, ranging from movie recommendations to product predictions and emotion recognition [12]–[14]. Such innovations have spurred researchers to increasingly apply DL in Twitter sentiment analysis, revolutionizing the field with enhanced accuracy and applicability.

Social media platforms have become a digital pulse for public opinion, serving as a vast repository of insights that are invaluable to industries across the spectrum, from marketing strategists to public policy makers. The sentiments expressed on these platforms provide a real-time barometer of public mood and opinion. Accurately capturing and interpreting this sentiment allows businesses and policymakers to make data-driven decisions that are responsive to the current climate, whether to pivot a marketing strategy, gauge the reception of a product, or assess the public's reception to policy changes.

The sheer volume of data produced on social media platforms, coupled with its dynamic and often ephemeral nature, presents a challenge that traditional data processing systems are ill-equipped to handle. This necessitates automated systems that are not only intelligent and capable of understanding the nuances of human language but also agile enough to process and analyze data in real time. Such systems could provide businesses and policymakers with the ability to respond to emerging trends and sentiments as they unfold, rather than in hindsight.

This work is squarely aimed at advancing the development of these automated systems. By harnessing the latest advancements in deep learning, we aim to refine the accuracy and

efficiency of sentiment analysis tools. The major objectives that we want to carry out for this project are given below:

- Utilize state-of-the-art deep learning models, such as Simple RNN, LSTM, Bi-directional LSTM, CNN, and BERT, that can effectively contextualize language.
- Study the findings from the models and compare their outcomes on Twitter data.

II. Literature Review

In light of the burgeoning significance of social media sentiment, this work aims to harness cutting-edge deep learning models to refine sentiment analysis tools for robust industry application. Alharbi et al. [15] leveraged CNN for Twitter sentiment analysis, achieving superior accuracy over traditional methods like LSTM, SVM, and KNN. This underscores the potential of deep neural networks in decoding complex sentiment expressions.

Building on this, Tam et al. [16] integrated CNN with Bi-LSTM to extract and contextualize features from tweets, demonstrating impressive accuracy across different datasets. Their work exemplifies the synergy between convolutional and recurrent neural networks in handling textual data.

Furthering the conversation, Chugh et al. [17] introduced an optimized DeepRNN model, which, combined with innovative optimization algorithms, achieved high accuracy in sentiment classification and information retrieval, highlighting the effectiveness of hybridized deep learning approaches.

Alamoudi et al. [18] explored aspect-based sentiment analysis using deep learning and various word embeddings, showcasing ALBERT's remarkable performance on binary classification tasks. This indicates the potential of transformer models in nuanced sentiment detection.

Tan et al. [19] presented a hybrid approach with RoBERTa and LSTM, addressing the challenges of temporal dependencies and data imbalance in sentiment analysis, again emphasizing the adaptability and precision of deep learning.

Lastly, Hasib et al. [20] applied DNN and CNN models to airline service tweets, achieving substantial precision and recall, which illustrates the practical application of deep learning in industry-specific sentiment analysis. Their work, along with that of Carvalho and Guedes [21], who developed novel term weighting schemes, and Pu et al. [22], who implemented an ensemble learning architecture, demonstrates a trajectory towards increasingly sophisticated and tailored sentiment analysis solutions.

Additionally, Chen, J et al. [23] utilized a three-way decision model for optimal feature selection, enhancing binary sentiment analysis's efficacy. This innovation reflects the ongoing evolution of feature extraction techniques in the field.

In conclusion, the collective contributions of these researchers provide a rich tapestry of methodologies and insights, reinforcing the main objective of this work: to advance automated systems for sentiment analysis using deep learning to meet the demands of the dynamic social media landscape.

III. Methodology

A. Traditional Deep Learning (DL) Models

Upon collection of the Twitter dataset, we embarked on the crucial pre-processing phase. This phase entailed several steps aimed at refining the dataset for optimal DL model performance. These steps included tokenization, the elimination of stopwords, stemming, and the correction of slang and acronyms. Additionally, we removed numerals, punctuation, and symbols, and transformed uppercase letters to lowercase to maintain consistency. Superfluous characters, URLs, hashtags, and user mentions were also excised from the dataset.

Following the pre-processing, we applied Word2Vec to model our word embeddings, setting the stage for the subsequent data input into our various models. This meticulous approach to pre-processing is designed to minimize noise and ensure that our DL models are primed for the most accurate sentiment classification possible. Figure 1shows a diagrammatic representation of the proposed system using basic deep learning models.

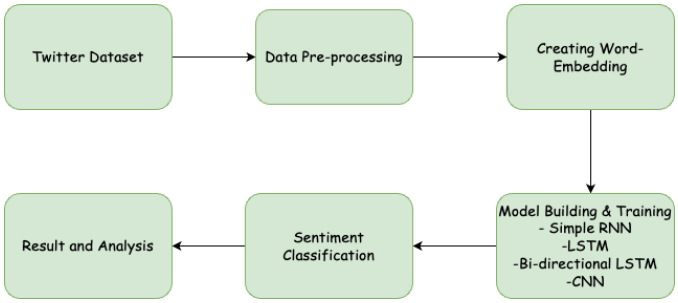

Fig. 1: System Architecture for Traditional DL Models

1) Dataset Description: In our study, we have employed a suite of deep learning (DL) techniques for the sentiment classification of tweets from the widely accessible Sentiment140 dataset. It contains 1,600,000 tweets extracted using the twitter API. The tweets have been annotated (0 = negative, 4 = positive) and they can be used to detect sentiment [25].

2) Tweets pre-processing: Pre-processing is converting the long data into short text to perform other processes such as classification, detecting unwanted news, sentiment analysis, etc., as Twitter users use different styles to post their tweets. Some may post the tweet in abbreviations, symbols, URLs, hashtags, and punctuation. Also, tweets may consist of emojis, emoticons, or stickers to express the user's sentiments and feelings. Sometimes the tweets may be in a hybrid form, such as by adding abbreviations, symbols, and URLs. Therefore, to further categorize the dataset, these kinds of symbols, abbreviations, and punctuations should be eliminated from the tweet. The features to be removed from the tweet dataset are tokenization, stopwords removal, stemming, slag and acronym correction, removal of numbers, punctuation and symbol removal, noise removal, URL, hashtags, replacing long characters, upper case to lower case, and lemmatization.

Model: "Sentiment_Model_BiLSTM"

Layer (type)	Output Shape	Param #
embedding_2 (Embedding)	(None, 60, 100)	6000000
bidirectional (Bidirectional)	(None, 60, 200)	160800
dropout_4 (Dropout)	(None, 60, 200)	0
bidirectional_1 (Bidirectional)	(None, 200)	240800
dropout_5 (Dropout)	(None, 200)	0
dense_5 (Dense)	(None, 64)	12864
dropout_6 (Dropout)	(None, 64)	0
dense_6 (Dense)	(None, 16)	1040
dropout_7 (Dropout)	(None, 16)	0
dense_7 (Dense)	(None, 1)	17

(a) Bidirectional LSTM

Model: "Sentiment_Model_LSTM"

Layer (type)	Output Shape	Param #
embedding_4 (Embedding)	(None, 60, 100)	6000000
lstm_4 (LSTM)	(None, 60, 100)	80400
dropout_8 (Dropout)	(None, 60, 100)	0
lstm_5 (LSTM)	(None, 100)	80400
dropout_9 (Dropout)	(None, 100)	0
dense_8 (Dense)	(None, 64)	6464
dropout_10 (Dropout)	(None, 64)	0
dense_9 (Dense)	(None, 16)	1040
dropout_11 (Dropout)	(None, 16)	0
dense_10 (Dense)	(None, 1)	17

(b) LSTM

Fig. 2: Model Architecture: Bi-LSTM Vs LSTM

3) Tokenization: Tokenization is splitting a text cluster into small words, symbols, phrases, and other meaningful forms known as tokens. These tokens are considered as input for further processing. Another important use of tokenization is that it can identify meaningful words. The tokenization challenge depends only on the type of language used. For example, in languages such as English and French, some words may be separated by white spaces. Other languages, such as Chinese and Thai words, are not separated. The tokenization process is carried out in the NLTK Python library. In this phase, the data is processed in three forms: convert the text document into word counts. Secondly, data cleansing and filtering occur, and finally, the document is split into tokens or words. For this project, we have set the vocab_length = 60000.

4) Embedding Matrix: An embedding matrix is a crucial component in the field of Natural Language Processing (NLP), specifically in tasks involving word representations. It's essentially a table where each row typically corresponds to a vector that represents a word within a vocabulary. These vectors capture semantic meaning and relationships between words in a lower-dimensional space compared to one-hot encoded vectors, which are high-dimensional and sparse.

Here's a detailed breakdown of the concept:

1) **Word Embeddings:** Word embeddings are learned representations of words where each word is mapped to a continuous vector. Semantically similar words have vectors that are close to each other in the vector space. The Word2Vec model is trained on the preprocessed tweets to learn word embeddings in this project.
2) **Matrix Structure:** The embedding matrix is typically initialized with random weights and has a size of [vocabulary_size x embedding_dimension], where vocabulary_size is the number of unique words in the corpus, and embedding_dimension is the size of the vector space in which words will be embedded. For our project, Embedding Matrix Shape is (60000, 100)

5) Model Configuration: Model building in the context of sentiment analysis of tweets involves creating various neural network architectures that are capable of understanding and classifying the sentiments expressed in the text. Each architecture has its own unique way of processing the data, and the choice of architecture can significantly affect the performance of the sentiment analysis. Here's an elaboration on the different architectures mentioned:

Simple RNN (Recurrent Neural Network): Simple RNNs are foundational to processing sequences, as they have memory capabilities that allow them to remember previous inputs. However, they are limited in their ability to handle long sequences due to the vanishing gradient problem, where the influence of inputs decreases over time.

LSTM (Long Short-Term Memory): LSTMs are a type of RNN that are designed to overcome the limitations of Simple RNNs by being able to learn long-term dependencies. They have a more complex structure with a memory cell and multiple gates (input, output, and forget) that regulate the flow of information, enabling them to remember and forget information selectively.

BiLSTM (Bidirectional Long Short-Term Memory): BiLSTMs are an extension of LSTMs that process data in two directions, forward and backward, which helps in capturing context from both directions. This bidirectional processing is particularly useful for understanding the meaning of words based on the surrounding words, improving the model's ability to understand context and nuance in language.

```
Model: "Sentiment_Model_CNN"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_4 (Embedding)     (None, 60, 100)           6000000

 conv1d_2 (Conv1D)           (None, 56, 128)           64128

 max_pooling1d_1 (MaxPoolin  (None, 28, 128)           0
 g1D)

 conv1d_3 (Conv1D)           (None, 26, 64)            24640

 global_max_pooling1d_1 (Gl  (None, 64)                0
 obalMaxPooling1D)

 dense_10 (Dense)            (None, 32)                2080

 dense_11 (Dense)            (None, 1)                 33

=================================================================
Total params: 6090881 (23.23 MB)
Trainable params: 90881 (355.00 KB)
Non-trainable params: 6000000 (22.89 MB)
```

(a) CNN

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 60, 100)           6000000

 simple_rnn (SimpleRNN)      (None, 100)               20100

 dense (Dense)               (None, 16)                1616

 dense_1 (Dense)             (None, 1)                 17

=================================================================
Total params: 6021733 (22.97 MB)
Trainable params: 21733 (84.89 KB)
Non-trainable params: 6000000 (22.89 MB)
_________________________________________________________________
```

(b) Simple RNN

Fig. 3: Model Architecture: CNN Vs RNN

CNN (Convolutional Neural Network): CNNs are typically associated with image processing but have been successfully applied to NLP tasks. In text analysis, CNNs can identify patterns or key phrases that signify sentiment by applying filters to the text, similar to how they identify edges and shapes in images. They are particularly good at picking up on local and position-invariant features in the data, which in the case of text, could be specific combinations of words or phrases that are indicative of sentiment.

To carry out this project's objectives, each model was configured with different numbers of layers, sizes of layers, types of activation functions (like ReLU, sigmoid), and methods for preventing overfitting, such as dropout or regularization techniques reffered architechture is 2 and 3. The configuration process involves fine-tuning these parameters to optimize model performance.

The main goal of experimenting with these different architectures is to determine which one is most effective at accurately classifying the sentiments of tweets. By testing various models, we compared their performance in terms of accuracy, speed, and their ability to handle the nuances of language expressed in tweets. Ultimately, the model that best captures the context and sentiment of the tweets, without overfitting to the training data, will be considered the most suitable for deployment in analyzing tweet sentiments.

B. Transformer Based DL Model: BERT

BERT, which stands for Bidirectional Encoder Representations from Transformers, is based on Transformers, a deep learning model in which every output element is connected to every input element, and the weightings between them are dynamically calculated based upon their connection; sample architecture is in figure 4.

Historically, language models could only read text input sequentially – either left-to-right or right-to-left – but couldn't do both at the same time. BERT is different because it is designed to read in both directions at once. This capability, enabled by the introduction of Transformers, is known as bidirectionality.

Preparing data for BERT involves a few steps that are designed to convert raw text into a format that BERT can understand. Here's an overview of the process:

1) **Tokenization:** BERT uses WordPiece tokenization. This means that words are broken down into smaller units (tokens) that can be found in its vocabulary. Each token is then replaced with its index from BERT's vocabulary. Special tokens are added: [CLS] at the beginning of each sentence (used for classification tasks) and [SEP] at the end of each sentence or to separate two sentences.
2) **Padding and Truncation:** BERT requires all sequences to be of the same length. If a sentence is shorter than the maximum sequence length, it is padded with zeros; if it's longer, it is truncated. The maximum sequence length is a hyperparameter typically set to 512 tokens for BERT. In this case, we have set the length 60.
3) **Attention Masks:** These masks tell BERT which tokens should be paid attention to and which should not (e.g., padding tokens). It's a sequence of 1s and 0s, with 1s for real tokens and 0s for padding tokens.
4) **Handling Special Cases:** BERT has special ways of handling out-of-vocabulary words, using sub-word tokenization. For example, the word "embeddings" might be split into "embed" and "##dings".

IV. Experimental Setup

Our models underwent a limited number of iterations, specifically between two to five epochs, which was deemed

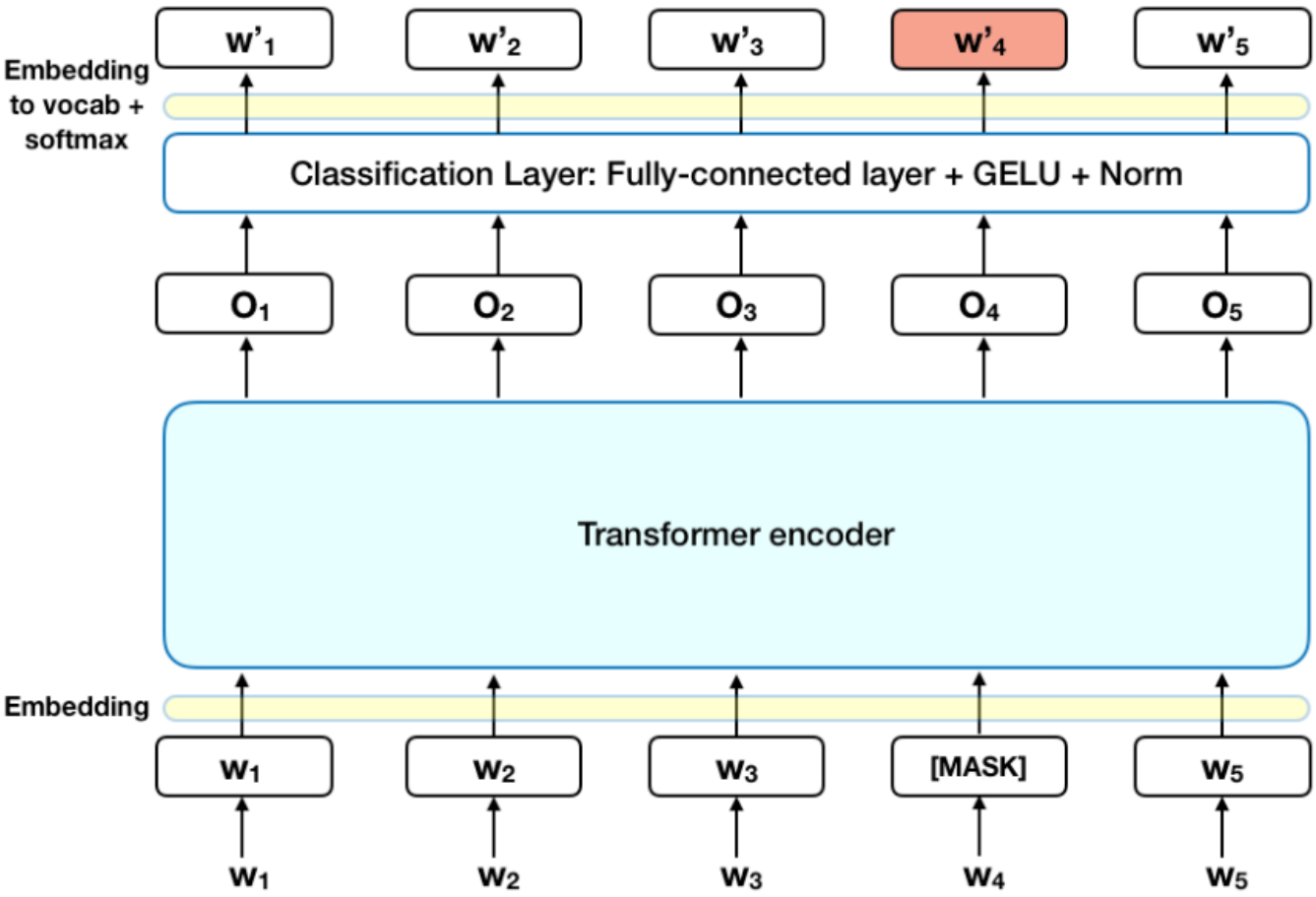

Fig. 4: Architecture of BERT

sufficient given the substantial size of our dataset comprising 1.6 million tweets. The data was divided into two segments: 80% for training purposes and the remaining 20% reserved for testing. In terms of duration, the traditional models completed their processing in 3 hours, whereas the BERT models demanded a more extensive time frame, exceeding 7 hours.

To facilitate these experiments, two distinct virtual environments were configured: one leveraging Python 3.12 for the conventional deep learning models, and the other utilizing Python 3.10 exclusively for BERT operations. The creation of traditional models was supported by TensorFlow's comprehensive toolkit, whereas the deployment of BERT was executed through PyTorch. The computational setup was powered by an Intel Xeon CPU with a 2.00GHz clock speed, supported by 12 GB of RAM, and was complemented by a Tesla K40c GPU, also with 12 GB of RAM.

V. Results

A. Confusion Matrix

A confusion matrix is a tool that helps to measure the performance of a classification algorithm. It is a table with two dimensions (”Actual” and ”Predicted”), and each dimension has two possible outcomes (here, ”Negative” and ”Positive”). The matrix compares the actual target values with those predicted by the machine learning model, which provides insight into how well the model is performing, with a focus on each class.

The confusion matrix typically contains the following four types of outcomes:

- True Positives (TP): The cases in which the model correctly predicted the positive class.
- True Negatives (TN): The cases in which the model correctly predicted the negative class.
- False Positives (FP): The cases in which the model incorrectly predicted the positive class (also known as Type I error).
- False Negatives (FN): The cases in which the model incorrectly predicted the negative class (also known as Type II error).

For sentiment analysis, a ”positive” might mean the tweet or text expresses a positive sentiment, while ”negative” could indicate a negative sentiment. Here's what each term represents in this context:

- True Negatives (TN): The number of correctly predicted negative sentiment tweets.

- True Positives (TP): The number of correctly predicted positive sentiment tweets.
- False Positives (FP): The number of negative sentiment tweets incorrectly identified as positive.
- False Negatives (FN): The number of positive sentiment tweets incorrectly identified as negative.

From the confusion matrix values, we can compute following evaluation matics:

- **Accuracy:** This is the overall correctness of the model and is calculated by

$$(TP + TN)/(TP + TN + FP + FN)$$

- **Precision (Positive Predictive Value):** This measures the accuracy of positive predictions. It's calculated by

$$TP/(TP + FP)$$

- **Recall (True Positive Rate or Sensitivity):** This measures the ability of the model to find all the positive samples. It's calculated by

$$TP/(TP + FN)$$

- **F1 Score:** This is the harmonic mean of precision and recall and is a single measure that balances both the concerns of precision and recall in one number.

B. Comparisons on Evaluation Metrics

- **Accuracy:** From figure 5, we can observe that BiLSTM has the highest accuracy (83.16%), followed by LSTM (82.96%), Simple RNN (81.59%), and CNN (80.87%). This suggests that BiLSTM and LSTM are more accurate overall.
- **True Positives:** LSTM has the highest percentage of true positives (42.14%), indicating it's slightly better at identifying positive sentiments than the other models.
- **True Negatives:** CNN has the highest percentage of true negatives (41.87%), suggesting it's slightly better at identifying negative sentiments than the others.
- **False Positives and Negatives:** Ideally, we want a model with low false positives and negatives. CNN has the highest false negatives (11.00%), indicating it misses more actual positive cases. BiLSTM has a balanced rate of false positives and negatives, contributing to its higher accuracy.

The F1 score would give us a better picture, but based on the provided accuracy and the distribution of true/false positives and negatives, the BiLSTM seems to be the most balanced and accurate model for this specific dataset and task. The LSTM also performs well and could be preferred in cases where identifying true positives is more critical. The CNN, despite having the highest true negative rate, falls behind due to its higher false-negative rate, indicating it might be overly cautious and miss positive sentiments. The Simple RNN, while generally good, doesn't excel in any particular area compared to the other models.

The BERT (Bidirectional Encoder Representations from Transformers) model achieving an accuracy of 87% suggests a significant performance edge over traditional deep learning (DL) models used for sentiment analysis on the same dataset. BERT's high accuracy indicates that it can generalize better on unseen data, which is a desirable trait in machine learning models. Another reason behind BERT's success is unlike traditional models that process data sequentially (left-to-right or right-to-left), BERT reads the entire sequence of words at once. This means it can capture information from both directions simultaneously, giving a fuller understanding of language structure.

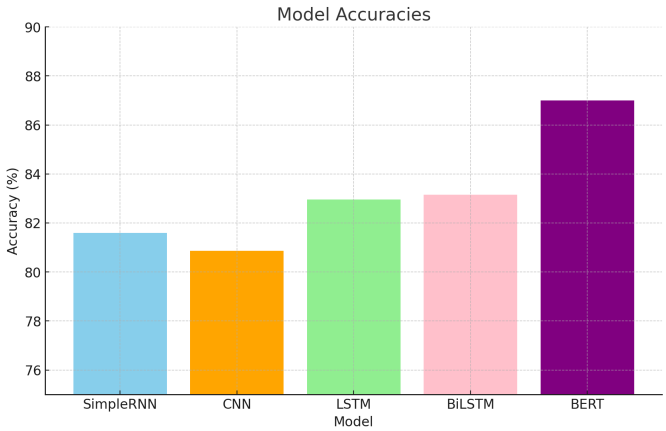

Fig. 5: Performance of the Models

VI. Conclusion and Future Works

The comparative analysis of sentiment analysis models reveals interesting insights into the performance of various neural network architectures. The BiLSTM model stands out with the highest accuracy, suggesting its strength in balancing the understanding of sequences from both directions, which is crucial in capturing the nuanced context of sentiments expressed in text data.

The LSTM model's higher true positive rate demonstrates its effectiveness in picking up positive sentiments. This might be particularly useful in scenarios where the identification of positive feedback is more important, such as analyzing customer reviews for positive experiences or monitoring positive reactions during a product launch on social media.

On the other hand, the CNN model's strength in identifying true negatives but weakness in false negatives indicates a tendency towards classifying sentiments as negative more often than they are. This conservative approach towards positive sentiment detection could be a drawback in applications where overlooking positive sentiments could lead to missed opportunities for leveraging positive customer feedback.

The Simple RNN, although the least performant among the architectures in this analysis, still offers valuable insights. Its simpler structure can be beneficial for faster iteration during the model development phase or when computational resources are a limiting factor.

However, the introduction of BERT into the comparison elevates the discussion to the next level of NLP performance.

BERT's significantly higher accuracy demonstrates the power of transformer-based models and their pre-trained representations, which provide a comprehensive understanding of language due to their attention mechanisms and bidirectionality.

The sequential data processing limitation of traditional models like RNN, LSTM, and CNN is overcome by BERT's ability to evaluate the entire input data all at once, leading to superior performance. This is particularly evident in sentiment analysis, where the context provided by surrounding words is paramount to understanding the sentiment of a specific word or phrase.

The use of BERT, despite its computational demands, seems justified when the goal is to achieve the highest possible accuracy. Its capacity to generalize and understand the complex structure of natural language makes it a robust choice for sentiment analysis and many other NLP tasks.

In conclusion, while BiLSTM and LSTM are strong contenders and may be preferred in resource-constrained environments or specific applications, BERT's advanced capabilities make it the superior choice for comprehensive sentiment analysis. Subsequent research endeavors may focus on refining BERT's efficiency and performance for even greater results, or on investigating other pre-trained languge models.

REFERENCES

[1] Saberi, Bilal, and Saidah Saad. "Sentiment analysis or opinion mining: A review." Int. J. Adv. Sci. Eng. Inf. Technol 7.5 (2017): 1660-1666.

[2] Medhat, Walaa, Ahmed Hassan, and Hoda Korashy. "Sentiment analysis algorithms and applications: A survey." Ain Shams engineering journal 5.4 (2014): 1093-1113.

[3] Drus, Zulfadzli, and Haliyana Khalid. "Sentiment analysis in social media and its application: Systematic literature review." Procedia Computer Science 161 (2019): 707-714.

[4] Zeglen, Eric, and Joseph Rosendale. "Increasing online information retention: analyzing the effects." Journal of Open, Flexible, and Distance Learning 22.1 (2018): 22-33.

[5] Qian, Yu, et al. "On detecting business event from the headlines and leads of massive online news articles." Information Processing & Management 56.6 (2019): 102086.

[6] Osatuyi, Babajide. "Information sharing on social media sites." Computers in Human Behavior 29.6 (2013): 2622-2631.

[7] Karami, Amir, et al. "Twitter and research: A systematic literature review through text mining." IEEE access 8 (2020): 67698-67717.

[8] Antonakaki, Despoina, Paraskevi Fragopoulou, and Sotiris Ioannidis. "A survey of Twitter research: Data model, graph structure, sentiment analysis and attacks." Expert Systems with Applications 164 (2021): 114006.

[9] Birjali, Marouane, Mohammed Kasri, and Abderrahim Beni-Hssane. "A comprehensive survey on sentiment analysis: Approaches, challenges and trends." Knowledge-Based Systems 226 (2021): 107134.

[10] Yadav, Nikhil, et al. "Twitter sentiment analysis using supervised machine learning." Intelligent data communication technologies and internet of things: Proceedings of ICICI 2020. Springer Singapore, 2021.

[11] Jain, Praphula Kumar, Rajendra Pamula, and Gautam Srivastava. "A systematic literature review on machine learning applications for consumer sentiment analysis using online reviews." Computer science review 41 (2021): 100413.

[12] Pandian, A. Pasumpon. "Performance evaluation and comparison using deep learning techniques in sentiment analysis." Journal of Soft Computing Paradigm 3.2 (2021): 123-134.

[13] Gandhi, Usha Devi, et al. "Sentiment analysis on twitter data by using convolutional neural network (CNN) and long short term memory (LSTM)." Wireless Personal Communications (2021): 1-10.

[14] Kaur, Harleen, et al. "A proposed sentiment analysis deep learning algorithm for analyzing COVID-19 tweets." Information Systems Frontiers (2021): 1-13.

[15] Alharbi, Ahmed Sulaiman M., and Elise de Doncker. "Twitter sentiment analysis with a deep neural network: An enhanced approach using user behavioral information." Cognitive Systems Research 54 (2019): 50-61.

[16] Tam, Sakirin, Rachid Ben Said, and Ö. Özgür Tanriöver. "A ConvBiLSTM deep learning model-based approach for Twitter sentiment classification." IEEE Access 9 (2021): 41283-41293.

[17] Chugh, Aarti, et al. "Spider monkey crow optimization algorithm with deep learning for sentiment classification and information retrieval." IEEE Access 9 (2021): 24249-24262.

[18] Alamoudi, Eman Saeed, and Norah Saleh Alghamdi. "Sentiment classification and aspect-based sentiment analysis on yelp reviews using deep learning and word embeddings." Journal of Decision Systems 30.2-3 (2021): 259-281.

[19] Tan, Kian Long, et al. "RoBERTa-LSTM: a hybrid model for sentiment analysis with transformer and recurrent neural network." IEEE Access 10 (2022): 21517-21525.

[20] Hasib, Khan Md, Md Ahsan Habib, Nurul Akter Towhid, Md Imran Hossain Showrov. A Novel Deep Learning based Sentiment Analysis of Twitter Data for US Airline Service. In 2021 International Conference on Information and Communication Technology for Sustainable Development (ICICT4SD), pp. 450–455. IEEE. 2021.

[21] Carvalho F. Guedes, GP. 2020. TF-IDFC-RF: a novel supervised term weighting scheme. arXiv preprint arXiv:2003.07193.

[22] Pu, Xiaomin, et al. "Sentiment analysis of online course evaluation based on a new ensemble deep learning mode: evidence from Chinese." Applied Sciences 11.23 (2021): 11313.

[23] Chen, Jie, et al. "A classified feature representation three-way decision model for sentiment analysis." Applied Intelligence (2022): 1-13.

[24] Jain, Deepak Kumar, et al. "An intelligent cognitive-inspired computing with big data analytics framework for sentiment analysis and classification." Information Processing & Management 59.1 (2022): 102758.

[25] Go, A., Bhayani, R. and Huang, L., 2009. Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford, 1(2009), p.12.