Enhancing Sentiment Analysis on Social Media with Novel Preprocessing Techniques

Khouloud Safi Eljil [1,2,*], Farid Nait-Abdesselam [3], Essia Hamouda [4], and Mohamed Hamdi [2]

[1] Computer Science Department, Université Paris Cité, France
[2] Higher School of Communications, University of Carthage, SUP'COM, Tunisia
[3] School of Science and Engineering, University of Missouri, Kansas City, USA
[4] Department of Information and Decision Sciences, California State University, San Bernardino, USA
*Correspondence: khouloud.safi-eljil@etu.u-paris.fr (K.S.E.)

***Abstract*—Sentiment analysis is a highly valuable tool, particularly in the realm of social media, as it enables us to understand the public's opinions regarding specific products or topics. However, analyzing short and unstructured texts like tweets can present significant challenges. This paper explores conventional Machine Learning (ML) approaches like Naive Bayes, Logistic Regression, and Support Vector Machine to analyze sentiment and compares them against Bidirectional Encoder Representations from Transformer (BERT). Moreover, we suggest a new preprocessing technique for sentiment analysis to enhance the effectiveness of these methods. Our findings demonstrate noteworthy enhancements in the performance of conventional ML models. Interestingly, our study reveals that BERT outperforms all aforementioned models, yielding an accuracy of about 94%, though incurring a high computational cost. Additionally, Logistic Regression performs well with a 90.35% accuracy rate. With respect to feature extraction, we showcase that combining unigram and bigram words provides a more thorough comprehension of negation, as opposed to solely relying on unigrams. Finally, we propose an approach for managing emoticons and emojis that has proven to be useful in the fields of sentiment analysis and sarcasm interpretation.**

***Keywords*—natural language processing, machine learning, feature extraction, social media, comparative analysis**

I. Introduction

Social networking services such as Facebook and Twitter have become increasingly popular in recent years. This is largely due to the platforms' ability to allow users to express themselves freely and openly, sharing their opinions, likes, and dislikes with a global audience [1]. As of May 2022, approximately 867 million tweets are sent per day [2]. Processing these overwhelming online customers' reviews and opinions is of interest to service providers/manufacturers and users. This fueled the need for Sentiment Analysis (SA), which is an approach in Natural Language Processing (NLP). SA studies the subjective information in an expression, including opinions, appraisals, emotions, and attitudes towards a topic, person, or entity. Since 2004, SA has rapidly gained momentum and has become a highly active area of research [3]. In the literature, three main approaches are used for SA: Machine Learning [4] (including deep learning), Lexicon-Based [5], and Hybrid approaches [6]. Among these, Machine Learning is the most widely adopted and established approach. Handling negation [7] remains one of the most difficult and unresolved challenges in sentiment analysis. For example, sentences like, "I couldn't be prouder", and "I am not proud" may be classified as having negative sentiment, even though the sentences express opposite sentiments. In the literature negation words (not, never, can't, etc.) are often included in stop-word lists and removed from the text during the pre-processing step [3], which we find unjustifiable in SA. Another challenge in sentiment analysis is detecting sarcasm, which involves using language that signifies the opposite to convey contempt. The sentence "Of course, I am happy to spend all my money to buy a mobile!" is an example of a sarcastic expression.

Extracting proper sentiment becomes even more complex when dealing with short and noisy social media texts, that have many peculiarities. In particular, tweets havemany distinctive features such as, the use of hashtags (e.g.,"#sad") to express a feeling; the use of the symbol "@"to mention another user in twitter (e.g., "@user123"); the use of emoticons like ":)" and emojis like "😔" which can change the tweet meaning. For example, "I am #speechless😌" has the opposite sentiment of "I am #speechless 💔". Unfortunately, most of the current research does not take those special symbols into consideration for SA [8].

In this study, we have designed and compared two approaches for SA, highlighting the main contributions as follows:

- We explore and compare the performance as well as the computational cost of traditional Machine Learning (ML) methods with Bidirectional Encoder Representations from Transformers (BERT) models.
- We propose an enhanced preprocessing stage that includes data cleaning, normalization, and emoticon and emoji handling to extract relevant

Manuscript received April 18, 2023; revised May 10, 2023; accepted July 13, 2023; published November 16, 2023.

features. This approach has led to an approximate 3% improvement in accuracy for traditional ML models.

- We show that handling emoticons and emojis significantly enhances sentiment analysis, especially in capturing sarcasm and ambiguous expressions.
- We observed that traditional ML models exhibited comparable performance to BERT models, but with a significantly lower computational cost.
- We demonstrate that incorporating both unigrams and bigrams can improve context understanding, particularly in cases involving negation.
- We show that traditional ML models have outperformed the current state-of-the-art approaches in SA.

This paper is organized as follows: Section II describes the proposed methodology for both approaches. In Section III, we compare and discuss the results of sentiment analysis. The literature in sentiment analysis is reviewed and compared with our work in Section IV. Finally, in Section V, we present our conclusions.

II. The Proposed Methodology

In this section, we discuss our proposed method, which consists of four phases (see Fig. 1): an extensive and novel data preprocessing of the tweets, feature extraction, model creation and training, model assessment and validation on a separate test dataset.

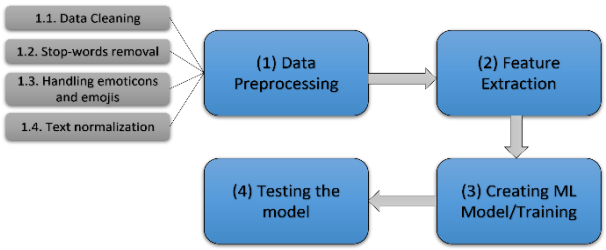

Figure 1. Model development process.

A. Dataset

We utilized the Kaggle Tweet Sentiment Extraction Dataset [9] for our study, which consists of two subsets, one for training and one for testing. The training dataset consists of 27,481 tweets and includes four fields: textID, Text, selected_text, and sentiment. We partitioned the training dataset into an 80% training dataset and a 20% validation dataset. The test dataset comprises 3534 tweets and three columns: textID, Text, and sentiment. The sentiment field in both datasets is the class attribute and has three labels: negative, neutral, or positive. As we focused on binary SA, we removed neutral tweets, resulting in a balanced datasetwith 47.6% negative and 52.4% positive tweets. The datasetdid not contain any null rows, but it was noisy (e.g., Ã¯Â¿)due to the nature of social media data, including repeated letters (e.g., wooow) and irrelevant parts such as "@users", emails, website links, and numbers.

B. Data Preprocessing

Data preprocessing is crucial in this study due to the noisy and unstructured nature of the text collected from social media websites. Additionally, tweets contain various language conventions and peculiarities. This phase includes four steps: data cleaning, stop-words removal, emoticon and emoji handling, and text normalization. The two relevant attributes in this research are the tweets' "text" as input and the "sentiment" as output. We mapped positive and negative sentiments to 1 and 0, respectively.

1) Data cleaning: In this step, regular expressions were used to clean the tweets by removing noisy data (e.g., Ã¯Â¿Â) and removing repeated letters occurring more than twice successively (e.g., "suppper" was replaced by "super"). We could have considered replacing a sequence of repeated characters by two characters instead of removing them, as done by [10], to differentiate between the regular usage of the word and the emphasized one. However, in our case, we did not have levels of positive and negative sentiments, such as extremely positive, positive, etc. Emails were simply removed, URL links were replaced with the word "link" and user mentions "@user" were replaced with "user_mention" to protect the privacy of Twitter users. Hashtags were removed but the words were kept, and words separated by underscores were replaced by whitespace (e.g., "#so_happy" was replaced by "so happy"). Numbers and punctuations were removed except for emoticons, which were handled separately and will be discussed later. Finally, all text was converted to lowercase.

2) Stop-words removal: Stop-words are commonly used words in any language that can be safely removed from textwithout sacrificing the meaning of the sentence. Removingthese words can help the model to focus on more relevant words. Although there is no well-defined list of stop-words,most researchers remove function words (i.e., words that serve to express grammatical relationships with other words within a sentence) such as "the", "at", and "which" [11]. In this study, we used the Natural Language Toolkit (NLTK) English stop-words in Python, which contains 179 tokens. We removed all the negative words (e.g., "mustn't", "wasn't", etc.) from the stop-words list as we believe that the negative form of words absolutely affects the sentiment of the tweet.

3) Emoticons and emojis handling: Despite their relevance to sentiment and their popularity on online platforms, especially social media websites, the literature on graphical emojis and their text-based precursors-emoticons-, is limited [12]. An emoticon, short for "emotion icon" is a combination of punctuation marks, numbers, and letters that represent a facial expression such as :-) for a smile or :-(for a frown, conveys the writer's intended tone or feelings [13]. Recently, emojis (😊), have joined the traditional text-based emoticons. Emoticons and emojis are crucial in communicating emotions that are difficult to express through words alone, such as sarcasm, which can be conveyed using ironic or exaggerated emoticons. However, processing emojis can be

particularly challenging due to their complex nature. Emojis are not standard characters but a combination of Unicode characters that can have multiple meanings and interpretations depending on the context and the cultural background of the users. Therefore, accurately identifying and analyzing the sentiment of a tweet that contains emojis requires sophisticated techniques that can account for the nuances and complexities of these visual symbols. In this research, we used the western-style list of emoticons from [14] and replaced each emoticon with its corresponding meaning using regular expressions. For example, we replaced :) with the word "smiley". To handle emojis, we employed the Emoji module, a Python package that represents each emoji by its name. For example, demojizin [1] 😍 yields: heart_eyes. We removed the leading and trailing colons as well as any underscores that occur between the words.

4) Text normalization: Text normalization is the process of reducing variations in word forms to their common original root or base form. This normalization simplifies the modeling process by decreasing the number of features, which can improve the performance of the model. To achieve text normalization, we utilized two techniques: lemmatization and stemming.

a) Lemmatization: Lemmatization involves mapping words to their base form (lemma) based on their dictionary definition and part of speech by employing vocabulary and morphological analysis of words. For example, the words "sang" and "sung" could be lemmatized to "sing".

b) Stemming: Stemming involves reducing words to their root form by removing suffixes and prefixes. It uses heuristic rules to strip suffixes according to a predefined list of derivational affixes. For example, the words "trouble", "troubled" and "troubles" could be stemmed to "troubl". This may result in misspelled words, but it is faster and simpler than lemmatization.

In this research we used an implementation of Porter algorithm for suffix stripping [15] and Wordnet Lemmatizer of NLTK for stemming, and lemmatization respectively.

C. Feature Extraction

Feature extraction is a crucial phase in sentiment analysis since the accuracy of the tweet classification depends on the features used as input. The first step in this phase is text tokenization, which involves breaking down a piece of text into smaller units, or tokens. Tokens can be words, characters, or sub-words [16]. These tokens are then converted into numerical features, commonly known as word embeddings in NLP, which the model can process efficiently. Fig. 2 illustrates the steps involved in feature extraction phase. We used the NLTK word tokenizer, which tokenizes at the word level. Remarkably, it also tokenizes contractions in an interesting way, such as representing "he's" as "he" and "'s". Moreover, for contractions with negations, like "haven't", it tokenizes it into "have" and "n't", which is advantageous, as will be discussed in Section III.

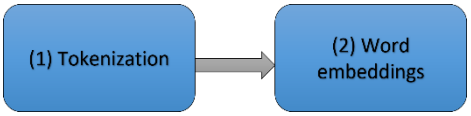

Figure 2. Features extraction phase.

There are two main types of word embeddings commonly discussed in literature: frequency-based embeddings and pre-trained word embeddings. In the following we will provide an overview of each type.

1) Frequency based embeddings: vectorize text by considering the frequency of word occurrences in the text or tweet. These embeddings are typically used with traditional NLP methods, such as count vector and Term Frequency-Inverse Document Frequency (TF-IDF).

a) Count vectorization: also known as bag of words (BoW), it is a matrix that contains all the distinct words in a document and their frequencies of occurrence. The order of the words is not important.

When preprocessing tweets in the training dataset, the vocabulary consisted of 18,527 distinct words, which were reduced to 11,644 unique words (unigram features). N-grams are continuous sequences of words or tokens in a document, where n is the number of those words. For example, the unigrams (n=1) of the sentence "I am not good" are ["I", "am", "not", "good"], while the bigrams (n=2) are ["I am", "am not", "not good"]. A combination of both unigrams and bigrams results in ["I", "am", "not", "good", "I am", "am not", "not good"]. We explored two types of word *n*-grams, unigram and both unigram and bigram, to capture the context of two contiguous words. This resulted in 75,846 features. There is a debate in the literature as to whether higher-order n-grams are better for sentiment classification. While higher-order *n*-grams may help in understanding the context, theycan result in a very sparse feature space that may not help ML algorithms in detecting patterns, and memory can also be a concern.

b) TF-IDF vectorization: is a statistical measure that reflects the importance of a word to a document in a collection of documents [17]. In this work, we created two separate matrices: one for unigram features and another for both unigram and bigram features. As a result, the number of features for only unigrams was 11,644, and the number of features for both unigrams and bigrams was 75,846.

2) Pre-trained word embeddings: word embedding is a technique used to represent text, in which words that have similar meanings are represented by similar vectors in a high-dimensional space. To obtain word embeddings, a neural network model is trained on a large corpus of text, and each distinct word in the corpus is represented with a corresponding vector in this space [18]. This approach is considered one of the key breakthroughs of deep learning in natural language processing. In this study, we explored BERT word embeddings.

[1]Converting graphical emoji to its meaning in text.

D. Machine Learning Models

In an effort to find the best model for sentiment analysis, we started by exploring traditional classifiers. We then turned to transformers, a cutting-edge deep learning algorithm that has achieved significant success in NLP.

1) Traditional ML models: To begin our sentiment analysis, we started with the Naïve Bayes (NB) classifier, which is a linear probabilistic model based on Bayes' theorem [19]. We chose NB due to its simplicity and popularity in sentiment analysis literature. Next, we investigated the use of Logistic Regression (LR) as a generalized linear model [20]. Finally, we explored Support Vector Machines (SVM), which is a non-probabilistic linear model capable of solving both linear and non-linear problems [21]. For all these traditional models, we used frequency-based word embeddings, specifically count-vectorization and TF-IDF, as inputs. Following the pre-processing steps outlined in Subsection II-B, we created four separate input matrices. Specifically, we extracted features using count vectorization twice: first with only unigram words, and then with both unigram and bigram features. We repeated this process using TF-IDF as well. To identify the best sentiment analysis model, we trained each of the three models (NB, LR, and SVM) with four different input features: Unigram Count Vectorization (UCV), Unigram and Bigram Count Vectorization (UBCV), Unigram TF-IDF (UTF-IDF), and Unigram and Bigram TF-IDF (UBTF-IDF). The performance results for each model and input feature combination are presented in Section III.

2) BERT models: A transformer [22] is a deep learning model that leverages the self-attention mechanism to assign different weights to each input feature based on its importance. This attention mechanism provides context for any position in the input sequence, allowing for simultaneous processing of the entire input. By considering all the surrounding words, the transformer enables BERT to better understand the meaning of a word in context. In this study, we have fine-tuned BERT [23], a pretrained bidirectional transformer model. BERT was originally trained on large unlabeled datasets, including the Books corpus (with 800 M words) and English Wikipedia (with 2,500 M words). These pretrained models can be easily fine-tuned in one of the downstream NLP tasks in what is known as transfer learning. Fig. 3 shows the architecture of the BERT model used for sentiment analysis. The first token of each sequence is always a special classification token [CLS], and the final hidden state corresponding to this token is used as the aggregate sequence representation for classification tasks [23].

We have examined three architectures of BERT: $\text{BERT}_{\text{BASE}}$, $\text{BERT}_{\text{LARGE}}$, and DistilBERT. DistilBERT is a distilled version of BERT that is smaller (40%) and faster (60%), while reported to maintain 97% of BERT's performance [24].

To preprocess the data in this analysis, we followed all the steps outlined in Subsection II-B except for lemmatization, stemming, and stop-words removal. With transformers, such normalization techniques are not necessary, as the BERT tokenizer performs word-piece splitting and provides proper word embeddings. Additionally, the attention mechanism in BERT eliminates the need for stop-words removal. We trained all three models for four epochs, which is the recommended number by the BERT team. We used Google Colab's GPU to train DistilBERT and $\text{BERT}_{\text{BASE}}$ with a batch size of 100 for both training and evaluation. For $\text{BERT}_{\text{LARGE}}$, we utilized Google Colab's TPU with eight cores, and used a batch size of 20 per core for both training and evaluation. The results of the models' performance are reported and discussed in the next section.

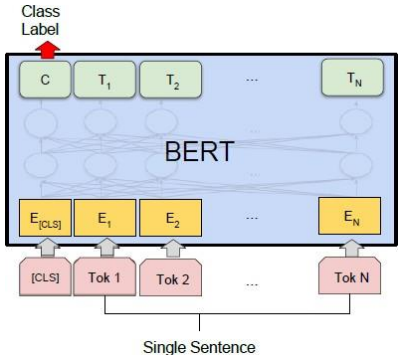

Figure 3. Sentiment classification using BERT [23].

III. Results and Discussion

A. Traditional ML Models

To compare and evaluate the performance of different ML models, we used four metrics [25]: accuracy (given by Eq. (1)), precision (given by Eq. (2)), recall (given by Eq. (3)), and F1-score (given by Eq. (4)), where TP, TN, FP and FN refer to "True Positive", "True Negative", "False Positive", and "False Negative" respectively. The results are presented in Table I, which compares the performance of three traditional ML models on the preprocessed test dataset, using various frequency word embeddings. All three models performed well, with accuracy ranging from 88.17% for Naïve Bayes with UCV word embeddings to 90.35% for LR with UBCV word embeddings. The training time for the different models was relatively short, ranging from 0.01 s to 99.21 s. Notably, Naïve Bayes had the shortest training time but also the least accuracy, while the best model was LR using both unigram and bigram count vectorization.

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (1)$$

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (2)$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (3)$$

$$\text{F1} - \text{score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

TABLE I. PERFORMANCE COMPARISON OF THE PROPOSED TRADITIONAL ML MODELS USING DIFFERENT FREQUENCY WORD EMBEDDINGS

Model	Word embeddings	Accuracy	Precision	Recall	F1-score	Training Time (sec)
Naïve Bayes	UCV	88.17	88.18	88.17	88.17	0.01
	UBCV	88.97	88.98	88.97	88.97	0.02
	UTF-IDF	87.50	87.55	87.39	87.45	0.02
	UBTF-IDF	88.69	88.82	88.54	88.63	0.03
LR	UCV	89.88	89.95	89.88	89.88	65.15
	UBCV	90.35	90.40	90.35	90.36	78.83
	UTF-IDF	89.83	89.86	89.83	89.83	15.40
	UBTF-IDF	89.92	89.93	89.92	89.93	62.53
SVM	UCV	89.35	89.55	89.35	89.36	10.81
	UBCV	89.21	89.39	89.21	89.22	30.98
	UTF-IDF	89.69	89.72	89.69	89.69	9.34
	UBTF-IDF	90.07	90.08	90.07	90.07	36.56

Results show that SVM's performance with unigram and bigram TF-IDF features was comparable to the best model. Additionally, using both unigram and bigram features yielded better results than using only unigrams in both count vectorization and TF-IDF across all models, with one exception. Specifically, the accuracy of SVM when using UCV was slightly better than when using both unigrams and bigrams (+0.14%). The input matrix size for both UCV and UTF-IF unigrams is 1.15 MB, while it is 2.12 MB for both unigrams and bigrams features. The size of the input matrix could be a concern for larger datasets or when considering higher n-grams. To assess the effect of preprocessing, we trained the models on the raw dataset (without preprocessing) using various frequency word embeddings. Fig. 4 shows the performance of the different models with and without preprocessing of the test datasets. Using the proposed preprocessing led to an improvement in performance for all models (around +3%). Additionally, the proposed preprocessing reduced the computational cost, with an average reduction in training time of 13.70 s.

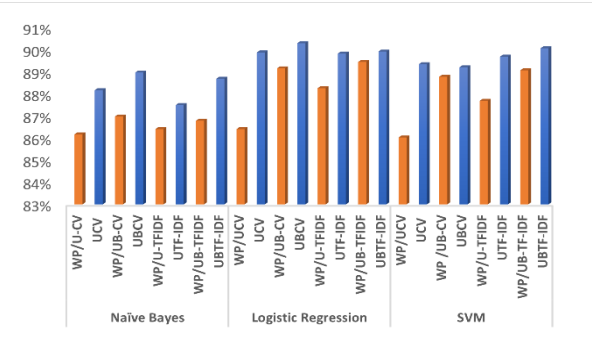

Figure 4. Accuracy comparison among proposed traditional ML models using different frequency word embeddings. The orange bars represent performance without preprocessing, while the blue bars represent performance using preprocessed test dataset.

Both UCV and UBCV count vectorization gave marginally better results than UTF-IDF and UBTF-IDF, respectively, when used with the NB and LR models. However, the opposite is true for the SVM model. The importance of TF-IDF is to minimize the weight of commonly occurring words that may not be relevant to the classification task. However, since we have already removed stop words during the preprocessing phase and the dataset is not specific to a particular domain or product, we do not have a problem of common words.

1) Most important features/words: We used the best-obtained model, namely the LR model, to obtain the most important words for negative and positive sentiments using only unigrams (Fig. 5) and both unigrams and bigrams count vectorization (Fig. 6). It is interesting to note that while using only unigrams, the model was able to identify the most important words for each sentiment correctly. However, using both unigrams and bigrams was better in associating the proper negative form of the words with the right sentiment. To elaborate, with unigrams (Fig. 5), the word "not" was identified as an important word for the negative sentiment, which is not always true. For example, "not bad" is a positive statement, as evident in (Fig. 6).

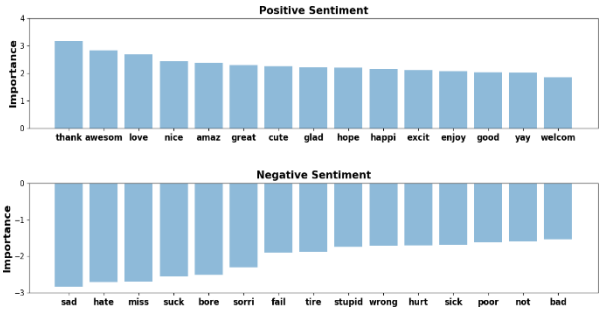

Figure 5. Most important words for negative and positive sentiments using unigram count vectorization.

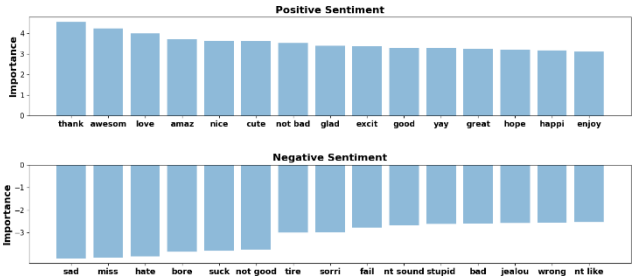

Figure 6. Most important words for negative and positive sentiments using LR model along unigram and bigram count vectorization.

2) Testing LR model with negative form sentences: We tested several sentences in the negative form using two LR models, one with UCV and the other with both unigrams and bigrams count vectorization. Table II shows the resulting sentiments, with correct sentiments underlined. These results demonstrate that using both unigrams and

bigrams count vectorization with the LR model is better at understanding context and determining the proper sentiment, especially for negative sentences.

TABLE II. TESTING NEGATIVE FORM SENTENCES USING LR MODEL WITH UNIGRAMS ONLY CV AND WITH BOTH UNIGRAMS AND BIGRAMS CV

Sentences in negative form	Using LR with UCV	Using LR with UBCV
I am not happy	positive	**negative**
No problem	negative	**positive**
Not bad, it is beautiful	negative	**positive**

B. BERT Models

Table III presents a summary of the performance results of three different versions of the BERT models, namely *DistilBERT*, $BERT_{BASE}$, and $BERT_{LARGE}$ on the test dataset. $BERT_{LARGE}$ achieved the highest accuracy of 94%. However, it came at a high computational cost, as it took about 30 minutes to train on eight cores Cloud TPU. On the other hand, *DistilBERT* and $BERT_{BASE}$ were trained in 3.45 min and 7.58 min, respectively, using a single core Cloud GPU. As expected, all three types of BERT models outperformed the best obtained traditional ML model (LR). Refer to Fig. 7 that compares the accuracy of the different BERT models with LR model on the test dataset.

TABLE III. PERFORMANCE COMPARISON OF THE THREE DIFFERENT TYPES OF BERT MODELS

Model	Accuracy	Precision	Recall	F1-score
DistilBERT	92.92	92.92	92.92	92.92
$BERT_{BASE}$	93.16	93.18	93.16	93.16
$BERT_{LARGE}$	94.01	94.01	94.01	94.01

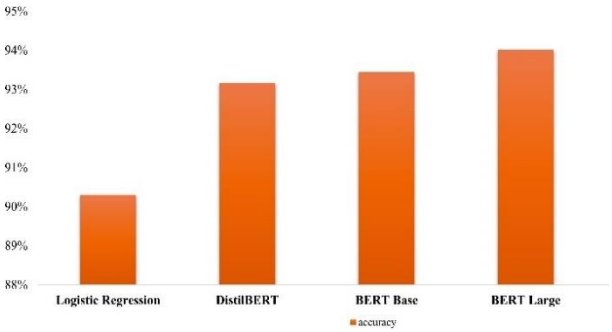

Figure 7. Comparison of accuracy: BERT models vs. Logistic Regression.

1) Negative form sentences and sarcasm: We conducted tests on some challenging sentences using the best obtained traditional ML model (LR) and the three types of BERT models. The resulting sentiments are presented in Table IV. As shown in Table IV, for the first sentence, all models predicted the correct sentiment, except LR, even though we used unigrams and bigrams CV. This is because the training dataset does not contain "not beautiful", but rather "beautiful", which appeared 97 times, with 92 of them classified as "positive". On the other hand, the different BERT models do not use frequency word embeddings but instead use rich contextualized pre-trained word embeddings. The fourth sentence was correctly classified by all models. Both *DistilBERT* and $BERT_{BASE}$ had the same misclassifications for the second, third, and fifth sentences. However, this was not the case for $BERT_{LARGE}$, where all sentences were correctly classified, except for the sarcasm (last sentence).

TABLE IV. AN EXAMPLE OF TESTING SOME SENTENCES USING THE BEST OBTAINED TRADITIONAL ML MODEL (LR), AND THE THREE TYPES OF BERT MODELS

Tweets	LR	DistilBERT	$BERT_{BASE}$	$BERT_{LARGE}$
1. It's not beautiful	positive	**negative**	**negative**	**negative**
2. I don't dislike horror movies	negative	negative	negative	**positive**
3. Couldn't be prouder	negative	negative	negative	**positive**
4. I am not proud	**negative**	**negative**	**negative**	**negative**
5. Of course I am happy to pay all my money to buy a mobile!	positive	positive	positive	positive

TABLE V. AN EXAMPLE OF TESTING SOME SENTENCES USING $BERT_{LARGE}$ MODEL WITH AND WITHOUT HANDLING EMOTICONS AND EMOJIS

Tweets	Without emoticons and emojis handling	With emoticons and emojis handling
Slept only 1 hour at night, woke up to no coffee, what a great day! :'(	positive	**negative**
Of course I am happy to pay all my money to buy a mobile! 😞	positive	**negative**
I am speechless 💔	**negative**	**negative**
I am speechless 😶	negative	**positive**

2) Emoticons and emojis handling: Our hypothesis is that handling emoticons and emojis can improve the model's ability to understand the context of sentences and detect sarcasm. To test this, we evaluated the $BERT_{LARGE}$ model, which was the best-performing model in both of our approaches, on some tweets that contain emojis and/or emoticons. If not handled properly, these special characters could be tokenized incorrectly or removed during preprocessing. For instance, BERT's tokenizer might replace the 💔 emoji with the unknown token [UNK] because it is not available in BERT original vocabulary. For emoticons, as they are mixtures of punctuations and numbers, if not handled properly, they will be removed during the preprocessing method. To address this issue, we apply our proposed method for handling emojis that replaces them with words that are present in BERT's vocabulary. For example, 😞 would be replaced by "disappointed face" which can then be tokenized into "disappointed" and "face" both of which have embeddings in BERT's vocabulary. Similarly, the emoticon: (would be replaced by "sad cry". The results of our evaluation are presented in Table V. As we can see, the proposed method

for handling emojis and emoticons has helped the model to detect sarcasm (sentences 1 and 2) and understand ambiguous tweets (sentences 3 and 4).

IV. A Comparative Related Work

Numerous traditional machine learning algorithms have been explored in the literature for SA, including NB, which is one of the most used models for text classification due to its simplicity and effectiveness. As a result, researchers often use it as a baseline model [26, 27]. In particular, Hasan *et al.* [27] designed a NB model that classifies Amazon reviews expressed in English and Bangla language without preprocessing for the English dataset. When the negation words were removed, the model achieved an accuracy of 86.7% on the English dataset. Surprisingly, the accuracy dropped to 85.7% when negations were considered, which may be attributed to using unigrams only. Our research involved investigating various types of word embeddings, encompassing both unigrams and bigrams as two consecutive words, in addition to conducting a thorough preprocessing stage. Consequently, we attained an accuracy of 88.97% for the Naive Bayes (NB) model.

SVM has been widely used by many researchers for sentiment analysis. Rana *et al.* [28] applied linear SVM and NB to classify movie reviews from the Internet Movie Database, which were categorized based on four genres. They found that linear SVM performed better than NB, with accuracy ranging from 72.50% to 87.50% for different movie genres. In our case, our SVM model with the proposed preprocessing achieved even higher accuracy, of 90.07%. Amrani *et al.* [29] used a hybrid approach that combined SVM and Random Forest algorithms to classify Amazon product reviews. They show that the hybrid approach outperforms NB and SVM models when implemented separately, achieving an accuracy of about 83.4%. Poornima *et al.* [30] used bigrams with three different models and found that LR outperformed both the multinomial NB and the SVM models, with an accuracy of 86.23%. Our proposed method, which involved an in-depth preprocessing stage and the use of both unigrams and bigrams with LR, outperformed all other traditional ML models and achieved even higher accuracy, with a score of 90.35%. Homaid *et al.* [31] analyzed the sentiment of Air-Traveller using VADER sentiment and LR and found that LR has outperformed VADER, with 87% accuracy compared to 59%. Ragothaman *et al.* [32] investigated the correlation between the tweets' sentiments and COVID-19 cases and deaths. Table VI presents a comparison between our proposed method, incorporating a novel preprocessing approach, and other research studies that have utilized traditional ML models for binary sentiment analysis. The results in the table demonstrate that our proposed method outperforms all the other models in the literature.

Recently after BERT was published as a revolutionary algorithm of deep learning in multiple tasks of NLP, some researchers chose to use BERT in SA. Sousa *et al.* [33] fine-tuned $BERT_{BASE}$ on a corpus of 582 financial news articles that were manually labeled as positive, negative, or neutral. They have compared $BERT_{BASE}$ with other models like NB, SVM, and textCNN. They concluded that $BERT_{BASE}$ has outperformed other models with 82.5% accuracy. Chiorrini *et al.* [34] investigated the use of $BERT_{BASE}$ model for both sentiment analysis (positive, negative, and neural) and emotion recognition of Twitter data. The reported results were interesting (92% accuracy), however the dataset used for SA is quite small (430 manually-annotated tweets) for training, validation, and test. Delobelle *et al*. [12] raised the issue of ill-equipped models to handle emojis. To overcome this, 3627 emoji tokens were added to the vocabulary of the BERT tokenizer. In a similar context, Lou *et al.* [35] proposed attention-based network models to improve emoji-based sentiment analysis on Chinese microblog posts.

TABLE VI. Comparing Our Method with the State-of-the-Art Methods for Binary Sentiment Analysis (POS, NEG) Using Traditional ML Models

Reference	Traditional ML Model	Best Obtained Accuracy
[27]	NB	86.7%
[28]	Linear SVM, NB	72.50% to 87.50% for the different movie genres
[29]	NB, SVM, and hybrid (NB + SVM)	83.4%
[30]	LR, NB, SVM	86.23%
[31]	VADER, LR	87%
Our method	**NB, LR, SVM**	**90.35%**

V. Conclusion

This paper investigates two approaches for sentiment analysis. Specifically, we examine conventional machine learning algorithms, including Naïve Bayes, LR, and SVM, and compare them to Bidirectional Encoder Representations from Transformer (*DistilBERT*, $BERT_{BASE}$, and $BERT_{LARGE}$). Our study suggests that traditional ML models can perform well with appropriate preprocessing and feature extraction. We propose a thorough preprocessing phase for sentiment analysis that significantly improves the results of traditional ML models. Regarding feature extraction, our results indicate that incorporating both unigrams and bigrams is more effective in capturing sentiment, particularly in the presence of negation. This approach enhances our understanding of sentiment compared to relying solely on unigrams. Furthermore, we propose a method for effectively managing emoticons and emojis, which has been effective in grasping the context of sentences and detecting sarcasm, thereby increasing accuracy.

Future studies can focus on analyzing tweets that contain a higher frequency of emoticons and emojis, as this can aid in improving the model's comprehension of their nuanced meanings. Additionally, training the model on a significantly larger dataset with an increased number of instances of sarcastic sentences and negation forms, can greatly enhance its proficiency in accurately recognizing sentiment in ambiguous sentences.

Conflict of Interest

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Khouloud Safi El Jil identified the research question and designed the study's methodology. She collected and analyzed the data, conducted experiments, and contributed to the interpretation of the findings. She diligently conducted a comprehensive literature review, ensuring the study's relevance within the academic context. Additionally, she took the lead in drafting and writing the manuscript, incorporating valuable insights and revisions from the advisors (Drs. Essia Hamouda, Farid Nait-Abdessalam, Mohamed Hamdi) and addressing reviewer comments. The advisors provided guidance, intellectual inputs, and supervision throughout the research process. They offered critical feedback on the research direction, refined the study's objectives, edited the manuscript, and facilitated access to necessary resources. All authors had approved the final version.

REFERENCES

[1] S. Ahmed, M. Pasquier, and G. Qadah, "Key issues in conducting sentiment analysis on arabic social media text," in *Proc. 2013 9th International Conference on Innovations in Information Technology (IIT)*, 2013, pp. 72–77.

[2] M. Yaqub. (2022). How many tweets per day 2022 (new data). [Online]. Available: https://www.renolon.com/number-of-tweets-per-day/

[3] M. Birjali, M. Kasri, and A. Beni-Hssane, "A comprehensive survey on sentiment analysis: Approaches, challenges and trends," *Knowledge-Based Systems*, vol. 226, 107134, 2021.

[4] A. Rashid and C.-Y. Huang, "Sentiment analysis on consumer reviews of amazon products," *International Journal of Computer Theory and Engineering*, vol. 13, no. 2, p. 7, 2021.

[5] H. A. Alhammi and K. Haddar, "Building a Libyan dialect lexicon-based sentiment analysis system using semantic orientation of adjective-adverb combinations," *Int. J. Comput. Theory Eng*, vol. 12, no. 6, pp. 145–150, 2020.

[6] Z. Madhoushi, A. R. Hamdan, and S. Zainudin, "Sentiment analysis techniques in recent works," in *Proc. 2015 Science and Information Conference (SAI)*, 2015, pp. 288–291.

[7] Sangeeta and N. G. Singh, "Review of factors affecting efficiency of twitter data sentiment analysis," *International Journal of Computer Theory and Engineering*, vol. 12, no. 1, pp. 53–58, 2020.

[8] P. Dandannavar, S. Mangalwede, and S. Deshpande, "Emoticons and their effects on sentiment analysis of twitter data," in *Proc. EAI International Conference on Big Data Innovation for Sustainable Cognitive Computing*, 2020, pp. 191–201.

[9] Tweet Sentiment Extraction. (2020). [Online]. Available: https://www.kaggle.com/c/tweet-sentiment-extraction

[10] A. Agarwal, B. Xie, I. Vovsha, O. Rambow, and R. J. Passonneau, "Sentiment analysis of twitter data," in *Proc. the Workshop on Language in Social Media (LSM 2011)*, 2011, pp. 30–38.

[11] C. Tan, L. Lee, J. Tang, L. Jiang, M. Zhou, and P. Li, "User-level sentiment analysis incorporating social networks," in *Proc. the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2011, pp. 1397–1405.

[12] P. Delobelle and B. Berendt, "Time to take emoji seriously: They vastly improve casual conversational models," arXiv preprint, arXiv:1910.13793, 2019.

[13] J. Daintith. (2018). Emoticon a dictionary of computing. [Online]. Available: https://www.encyclopedia.com/science-and-technology/computers-and-electrical-engineering/computers-and-computing/emoticon#EMOTICON

[14] Wikipedia. (2022). List of emoticons. [Online]. Available: https://en.wikipedia.org/wiki/List_of_emoticons

[15] M. F. Porter, "An algorithm for suffix stripping," *Program*, 1980.

[16] L. A. Mullen, K. Benoit, O. Keyes, D. Selivanov, and J. Arnold, "Fast, consistent tokenization of natural language text," *Journal of Open-Source Software*, vol. 3, no. 23, p. 655, 2018.

[17] A. Aizawa, "An information-theoretic perspective of TF–IDF measures," *Information Processing & Management*, vol. 39, no. 1, pp. 45–65, 2003.

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

[19] I. Rish, *et al.*, "An empirical study of the naive bayes classifier," in *Proc. IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence*, vol. 3, no. 22, 2001, pp. 41–46.

[20] F. E. Harrell *et al.*, *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*, Springer, 2001.

[21] N. Cristianini, J. Shawe-Taylor *et al.*, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press, 2000.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[23] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pretraining of deep bidirectional transformers for language understanding," arXiv preprint, arXiv:1810.04805, 2018.

[24] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter," arXiv preprint, arXiv:1910.01108, 2019.

[25] N. Japkowicz and M. Shah, *Evaluating Learning Algorithms: A Classification Perspective*, Cambridge University Press, 2011.

[26] X. Shao, C.-S. Kim, and K. D. Ryul, "A study on customers sentiment analysis based on big data using twitter data," *International Journal of Computer Theory and Engineering*, vol. 11, no. 1, pp. 11–14, 2019.

[27] K. A. Hasan, M. S. Sabuj, and Z. Afrin, "Opinion mining using naive bayes," in *Proc. 2015 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE)*, 2015.

[28] S. Rana and A. Singh, "Comparative analysis of sentiment orientation using SVM and naive bayes techniques," in *Proc. 2016 2nd International Conference on Next Generation Computing Technologies (NGCT)*, 2016, pp. 106–111.

[29] Y. Al-Amrani, M. Lazaar, and K. E. El-Kadiri, "Random Forest and support vector machine-based hybrid approach to sentiment analysis," *Procedia Computer Science*, vol. 127, pp. 511–520, 2018.

[30] A. Poornima and K. S. Priya, "A comparative sentiment analysis of sentence embedding using machine learning techniques," in *Proc. 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2020, pp. 493–496.

[31] M. S. Homaid, D. B. Bisandu, I. Moulitsas, and K. Jenkins, "Analysing the sentiment of air-traveller: A comparative analysis," *International Journal of Computer Theory and Engineering*, vol. 14, no. 2, pp. 48–53, 2022.

[32] A. Ragothaman and C.-Y. Huang, "Sentiment analysis on covid-19 twitter data," *International Journal of Computer Theory and Engineering*, vol. 13, no. 4, pp. 100–107, 2021.

[33] M. G. Sousa, K. Sakiyama, L. de Souza Rodrigues, P. H. Moraes, E. R. Fernandes, and E. T. Matsubara, "Bert for stock market sentiment analysis," in *Proc. 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, 2019, pp. 1597–1601.

[34] A. Chiorrini, C. Diamantini, A. Mircoli, and D. Potena, "Emotion and sentiment analysis of tweets using BERT," in *Proc. EDBT/ICDT Workshops*, 2021.

[35] Y. Lou, Y. Zhang, F. Li, T. Qian, and D. Ji, "Emoji-based sentiment analysis using attention networks," *ACM Transactions on Asian and Low-Resource language Information Processing (TALLIP)*, vol. 19, no. 5, pp. 1–13, 2020.