A COMPARISON OF ML APPROACHES ON SENTIMENT ANALYSIS, BASED ON ONTOLOGIES, SARCASM AND SUBJECTIVITY DETECTIONS IN THE CASE STUDY OF US ELECTIONS.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A COMPARISON OF ML APPROACHES ON SENTIMENT ANALYSIS, BASED ON ONTOLOGIES, SARCASM AND SUBJECTIVITY DETECTIONS IN THE CASE STUDY OF US ELECTIONS.

[1]IHAB MOUDHICH, [2]ABDELHADI FENNAN

[1] Abdelmalek Essaadi university, faculty of sciences and techniques, LIST Department of Computer Science, Tangier, Morocco

[2] Abdelmalek Essaadi university, faculty of sciences and techniques, LIST Department of Computer Science, Tangier, Morocco

E-mail: [1]ihab.moudhich@gmail.com, [2]afennan@gmail.com

ABSTRACT

Nowadays, Twitter has become one of the most excellent tools that give people the power to express their emotions. And also, to interact with other ordinary or political people. According to The Verge, as known as the American technology news website, more than 166 million users are using Twitter every day; this thing made Twitter one of the largest news sources and one of the places where most of the politicians publish their opinions or their thoughts. Sentiment analysis or opinion mining is a method that is used to understand the user's behavior based on their feelings in a given text, which can help to get a global idea of the expected outcomes of USA elections.

Our research is based on extracting the data and analyzing tweets' sentiment to predict the USA elections results. As we know, most Americans use Twitter to interact with each other to explain their opinions and thoughts about the subjects related to their country. Also, the hashtag system on Twitter makes it easy to help people to interact and go viral.

We also included other variables to make a significant comparison of our results, such as detecting sarcasm and subjectivity in a tweet. Also, we used two machine learning approaches: First known as Long Short-Term Memory (LSTM). The second is Bidirectional Encoder Representations from Transformers (BERT). In this work, we used more than 500,000 tweets to get a significant result. Moreover, our developed Framework consists of 5 steps: First, collecting data based on ontologies that we defined. Second, text pre-processing to clean data. Third, predicting subjectivity and sarcasm in a tweet. Fourth applying the two cited approaches to get the sentiment. And the last one is visualizing and analyzing the results.

Keywords: *Sentiment Analysis, Machine Learning, Ontology, LSTM, BERT, Sarcasm Detection, Subjectivity Detection.*

1. INTRODUCTION

Twitter has become one of the largest traffic sources that everyone uses to express their emotions by just writing a few words. Due to the large amount of data provided by Twitter, researchers started to notice that if they use Twitter [1], they will get benefit from such massive valuable data to expand their works [2].

As known as opinion mining, Sentiment analysis is considered a field within Natural Language processing [3]. It is used for identification if a given text is labeled as positive, neutral, or negative. It has become popular for researchers to work with sentiment analysis and machine learning approaches by applying ontologies to point directly to valuable data and get the best accuracy for good results.

Due to the ontologies [4], researchers' today's work on merging the ontology-based and sentiment analysis approaches, which makes them capable of defining the concepts and the relations between them to sort with a structure of ontological knowledge.

In this work, we proposed two methods to analyze US elections data, which we collected from Twitter based on our ontologies that we defined before. We moved to the text pre-processing phase to clean our data from noises. Also, we applied two different machine learning approaches, Long Term-Short Memory (LSTM) [5] and Bidirectional Encoder Representations from Transformers (BERT) [6]. We must also mention that we included other variables such as subjectivity and sarcasm detections to get multiple outputs to be compared with our previous work. We tested two different approaches to train our models for these variables, first using Long Term-Short Memory and second, using Support Vector Machine. We chose to continue with the one who will give us good accuracy.

The paper is organized into six sections. In section 1, we mentioned examples of related work that has a relation with our work. In section 2, we introduced the concept that we used in this work. Then in section 3, we described our proposed work with its different phases. Then in section 5, we presented our results, then in section 6, we moved to conclude our paper.

In this paper, we worked with BERT and LSTM approaches based on ontologies. To give our work a new challenge to go with multiple parameters that do not exist in the other experiments. Also, we apply new variables -such as sarcasm and subjectivity detections- to our models to see their impact on our results, which can give us multiple overviews in the analysis phase of our study.

In this work, we aim to compare the LSTM and BERT approaches. To understand which method gives a good accuracy between the two models. So, that we can improve it in our new experiment. Also, we must mention that our experiment will be compared with other available sources.

In this paper, we aim to provide a new idea of how we can use these two approaches, in sentiment analysis fields along with applying ontologies. So, we try to give a clear overview of them. And to explain all the steps that we made. To make it easy for other researchers to continue on the same path as we did. Also, in our work, we based on multiple previous studies that we will cite in each paragraph to give a big source of information.

2. RELATED WORK

A proposed system to analyze a text's sentiment using a pre-trained BERT model was presented by Manish et al [7]. They devised their architecture into four stages. First, they started with text pre-processing by applying the canonicalization, which means they removed all the digits, punctuation, and symbols. And then, they applied the tokenization by using a Word-Piece tokenizer [8]. Second, they moved to compute the sequence embedding from BERT. For the third stage, they used dropout to prevent overfitting. And for the last one, they used the SoftMax classification layer to classify labels to whom the text belongs.

Another system based on BERT was proposed by Mickel Hoang et al. [9]. They implemented three types of models. The first one is the aspect classification model, which used the sentence pair classification from BERT to predict that either an aspect is related to a text or not. Second, A sentiment polarity classification model was implemented to predict that either a sentiment label is positive or negative for a given text. And for the last one, there is a combination between the two previous methods to get an output that shows if the aspect is related and to which category it belongs.

The architecture of the system based on Brexit data was proposed by Moudhich Ihab et al [10]. In their work, they proposed a solution to analyze the Brexit data. They started by defining the ontologies after studying all the parliament parties and articles that explain and talk about Brexit. After that, they pre-processed their data after getting it from Twitter to make it clear from noises. To simplify the prediction of sentiment in a giving tweet, they compared an LSTM and a lexicon-based method to see who will give good accuracy. Also, we can conclude that LSTM worked well for them due to the situation of Brexit today.

Ontologies have a major impact in the sentiment analyses field, as Efstratios Kontopoulos et al. [11] mentioned in their work. They created two types of domain ontologies. The first one is the Formal Concept Analysis (FCA), which is considered a mathematical data analysis theory, and it is often used in Knowledge Representation and Information Management fields. It is based on building concepts, which have two sets, the extension, and intension. The second one is ontology learning, also known as ontology generation. It is based on creating ontology automatically by extracting relations and concepts from a given data set.

A comparison between different machine learning approaches has been made by Merfat el al. [12]. In this work, they provided different results of the sentiment analysis of Arabic Twitter data. Their Framework consists of six main phases: Dataset that determine the details of its content, Transformation to convert the data to the format which is ready for processing, pre-processing to normalize and clean data, Word vector representation to turn the data into vectors, classification and evaluation to evaluate their work. Moreover, they tested three machine learning algorithms, such as Support vector machine, Naïve Bayes, Decision Tree.

The research proposed by Efthymios Kouloumpis el al. [13] presents an evaluation of training data with and without hashtags and emoticons, further the evaluation of the effectiveness of the features from sections. They also proved in their paper that using hashtags to collect training data is useful and returns a good result. They also mentioned that they used different features in their classification experiences, such as unigrams, bigrams, and feature representing information.

A framework for sentiment analysis was proposed by Hamid Bagheri et al. [14]. Their work is based on extracting the sentiment from tweets. This work starts with downloading the sentiment dictionary, then getting the Twitter testing data, then cleaning and tokenizing the data, finally calculating the sentiment that has a big impact on the tweet. It means they calculated the number of positive and negative words within a tweet. The tweet sentiment was the biggest, and then they analyzed their results.

A subjectivity detection research was proposed by Luciano Barbosa et al. [15]. This Framework was started by collecting data from three sources: Twendz, Twitter sentiment, and TweetFeel. Then they moved to categorize a tweet by implementing a 2-step sentiment detection framework. The First is based on subjectivity detection, and the second on polarity detection: positive and negative. Also, they mentioned that they added the proper subjectivity -strong and weak subjectivity-.

Sarcasm detection has an important role in augmenting accuracy when we classify a text with positive or negative labels. An architecture was proposed by Yessi Yunitasari et al. [16]. In their work, they used an Indonesian tweet that was collected from Twitter. Then they pre-processed the data. After that, they extracted sarcasm features and applied a sarcasm detection algorithm. Finally, they analyzed their results. They also improved their model- if a tweet got a positive label and sarcasm words, they changed their label to negative.

A comparison between CNN-LSTM [17] model and regression-based, NN-based, and lexicon-based methods was proposed by Jin Wang et al. [18]. They used Stanford Sentiment Treebank (SST) and Chinese Valence-Arousal Texts (CVAT) in these experiments. They also used two lexicon-based methods, such as weighted geometric mean (WGM) and weighted arithmetic means (WAM), along with regression-based methods such as maximum values regression and average values regression. Their results show that WGM gives a better result than the WAM in the lexicon-based method. Also, the CNN-LSTM outperformed the NN-Based method.

Another work of LSTM was proposed by Yukun Ma et al. [19]. They evaluated their method with two datasets, SentiHood and Subset of Semeval. Also, they used two methods of sub-tasks of targeted ABSA: aspect-based sentiment classification and aspect categorization. In the final step, they compared their results with the other existing results to analyze their Framework's performances.

Research presented by Sporia et al. [20] demonstrates the sentiment and emotion clues. They also supposed that the user's personality is an important factor for their Framework about sarcasm detection. They created three different models: sentiment, emotion, and personality models. For the first two models, they worked with a Convolutional neural network known as CNN. For the personality model, they applied five personality types, i.e.., openness, conscientiousness Extraversion, Agreeableness, and Neuroticism. Also, they experimented with three types of datasets—the first dataset based on collecting tweets using sarcasm hashtag. For the second dataset, they worked with the English dataset from Ptacek et al. [21]. And the last one is based on the Sarcasm detector. Their pre-processing methods have two-steps: first, by removing all users, URL, hashtag. And for the second, they used NLTK [22] Twitter tokenizer to ensure proper tokenization.

3. Main Concepts

3.1 Sentiment Analysis

This is the process of extracting the opinion in a given text by applying different methods and approaches to natural language techniques, whether based on machine learning or based on lexicon-based approaches.

To analyze a text that users post on social media, we should use different sentiment analysis techniques. And that is considered as a classification problem. In most cases, the sentiment categorization is divided into three types: negative, neutral, positive. We must also mention that there are three types to extract a sentiment in a given text, either by using machine learning, lexicon-based approaches, or hybrid sentiment analysis approach.

In general, each sentiment analysis study is based on these essential steps, starting with collecting data, cleaning this data, apply one of the methods or approaches to detect a sentiment on a text, and finally, a visualization part to compare and analyze the result that the Framework got. Also, one of the challenges is to increase the accuracy of the results.

3.2 Ontology

Ontologies are based on knowing how the concepts are related to each other. In every case, we should study a subject and define its categories with its corresponding set of concepts.

Ontologies aim to give knowledge about a specific domain so that the researchers and developers could understand it. Also, it can be used to explain the relationship between the entities within a domain.

An ontology can also be applied to one domain because every domain has its case and definition.

3.3 Long-short term memory

Long-short term memory, known as LSTM, was introduced by Horchreiter and Schmidhuber in 1997. It's a type of recurrent neural network. It's based on remembering the information for a long period to avoid dependency problems. To apply LSTM, we should define what type of information we are going to use, which information will be suitable for our system, and what will be the output.

In recurrent neural networks known as RRN, a module will have a straight forward base structure, such as tanh with a single layer. On the other hand, we have the LSTM module with a different structure with four layers to interact differently.

3.4 Bidirectional Encoder Representations from Transformers

Bidirectional Encoder Representations from Transformers, known as BERT, is a natural language processing (NLP) based on Transformer, developed by Google in 2018 by Jacob.

Bert's model has two pre-trained types. The first one is the BERT (base) model with 12 layers, 768 hidden, and 110M parameters of neural network architecture. The second one is a BERT (large) model with 24 layers, 1024 hidden, and 340M parameter neural network architecture. These two models were trained on the BooksCorpus with more than 800M words and 2500M words of the English Wikipedia.

BERT can learn contextual relations between words in a text. A transformer has two separate mechanisms, an encoder that reads the text input and a decoder that predicts a text. A transformer reads the whole sequence of words at once that's why it has more accuracy than the read of text input sequentially (left-to-right or right-to-left).
A 15% of words in sequence are changed with [MASK] token, so it makes the BERT model able to predict the masked words based on the provided context. Each prediction of the output requires: Adding a classification layer, multiplying the output vectors by the embedding matrix, and then calculating the probability of words by SoftMax.
The loss function of BERT takes only the prediction of the masked values, ignoring the rest. And that gives the model slow coverage ability.

In BERT training, the model got a pair of sentences as input, and he learned to predict if the second sentence is the subsequent sentence in the original document. In each sentence, the [CLS] is inserted at the beginning and the [SEP] token at the end.

To conclude, a BERT can be used in different language tasks by adding a small layer to the core model. In classification tasks like sentiment analysis, they are treated as we treat the next sentence classification, by just adding a classification layer to the top of the transformer output.

3.5 Text pre-processing

Comparison research on text pre-processing has been made by Zhao et al. [23]. They worked on replacing negative mentions, such as replacing "can't" to "can not," etc., removing URLs, reverting words to their originals like "cooool" to "cool", removing numbers and stop words, expanding acronyms to their original. Then they gave the results of their experiment by showing the six pre-processing methods that they applied.

4. SENTIMENT ANALYSIS FRAMEWORK

This section will explain the methods and approaches that we used in this paper, starting with explaining our ontology, the pre-processing algorithm that we used, and the different machine learning approaches that we applied in our research. We will also give an overview of the variables that we added to our architecture to analyze their effect on the results. Also, we will give our framework a name: "SAF, sentiment analysis framework".

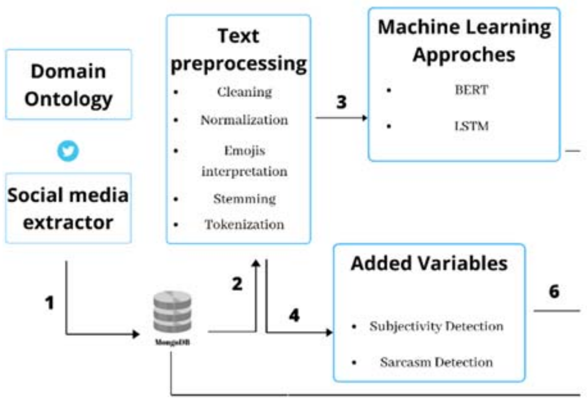

Figure 1: Global architecture of the proposed, sentiment analysis framework

As we can see in Figure 1, this a global architecture for our case study of American elections.

4.1 Ontology

2020 is the year of the United States elections, and for this time, it takes a huge intension from all the American people because every vote means a huge chance to one of the candidates to become a new president of the United States. And as we know, the competition now is between Donald Trump from the republican party and Joe Biden from the Democratic party.

As a first step, we started by studying the United States elections case and different parties' participants. We gathered politicians and parties' usernames and their used hashtags. We collected data using hashtags related to each side of the United States elections, such as "DonaldTrump", "JoeBiden".

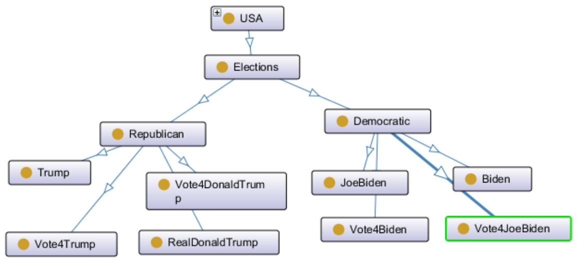

Figure 2: United State Elections Domain's Ontology

As we see in Figure 2, we used different keywords to describe our ontology.

4.2 Data Collection

In this step, we started collecting data based on Twitter API using Tweepy package with python language.

Our Script worked with collecting tweets based on hashtags and keywords that we defined with ontologies to ensure that we have the relevant data about United States elections.
We must also mention that all extracted tweets must have a keyword of election and the other keywords about one of the candidates, like "In this US elections, I will vote for #DonaledTrump."

In this case study of the united state elections, we collected about 500,000 tweets.

4.3 Text Preprocessing

In this step, we worked on transforming the previously collected data to noiseless data with a clean format. We thought this step is crucial to ameliorate our results' accuracy and fit our models with clean data to make it easy to learn and sort with a clean model and a good accuracy.

Before we started working on the pre-processing phase, we began by processing our collected data. That we used to train our different models. Also, we tried to make an additional test to find the necessary functions that we will apply without reducing our models' accuracy or performance.

After the first step, we sorted with functions that we applied to our tweets. We started by removing any existing URLs, numbers, screen names, or whitespaces. Next, we started contracting (e.g., Didn't become did not), along with the emojis' substitute to their original meaning. We also changed any existing abbreviations (e.g., LOL became Laughing Out Loud). Also, we moved to remove any existing stop words. After we tried to test different training models with stop words and without it, we found that the stop words do not affect the performance of our model-. Then we applied a lemmatization that consists of returning the word to its canonical form. And finally, we normalized the entire tweet into lower case.

Table 1: Describe the pre-processing steps

```
Algorithm: PreprocessingForTweets

Input: text
Variables: text
Output: clean_text
    1.  Text ← removeSpaces(text)
    2.  Text ← removeScreenName(Text)
    3.  Text ← removeNumbers(Text)
    4.  Text ← removeUrls(Text)
    5.  Text ← changeAbbreviation (Text)
    6.  Text ← emojisTransformation(Text)
    7.  Text ← RemoveStopWords(Text)
    8.  Text ← Lemmatization (Text)
    9.  Text ← lowerCase(Text)
    10. return clean_tweet
```

4.4 Datasets

In this step, we will give an idea about where we got our dataset to train our models and some references to make it easy for everyone to work with the same datasets that we used.

4.4.1 Dataset used for LSTM and BERT models

To train our LSTM and BERT models, we used the IMDB dataset. IMDB is considered a famous movie website, and it has labeled reviews - positive and negative-.

The IMDB dataset contains about 50000 movie reviews, with 25000 reviews labeled as positive and the rest as negative.

There are some other datasets, such as the Twitter dataset. But we chose to stuck with IMDB because we got an excellent accuracy than the Twitter dataset.

4.4.2 Dataset used for Sarcasm detection model

To Train our sarcasm model, we used News Headlines Dataset. Many researchers tried to collect sarcasm data from Twitter based on the hashtag with the word sarcasm (e.g., #sarcasm), but this dataset was noisy and hard to pre-process. Our dataset that we used is collected from two news websites, TheOnion and HuffPost. The first one has sarcastic data, and for the second one, we can find non-sarcastic data.

This dataset has many advantages. The headlines are written by professionals, which means we can't find spelling mistakes or informal usages. Also, we got high-quality labels because TheOnion is based on publishing sarcastic news.

This dataset contains three attributes. A headline attribute with a text, the second attribute is is_sarcastic has the information if the headline is sarcastic or not, and the last one has the article link.

4.4.3 Dataset used for Subjectivity detection model

To train our subjectivity model, we used a dataset from Cornell. This dataset has two files. The first one has 5000 subjectivity sentences, whereas the second one has 5000 objectivity sentences.

4.5 Experiments between SVM and LSTM for the variable's models

In these experiments, we tried to find the best algorithm to train the models for the variables that we used- Sarcasm and subjectivity detection-.

At first, we started by pre-processing our datasets to ensure that everything is cleaned. Second, we moved to train our data with Support Vector Machine known as SVM, and the same thing with Long-short term memory approach.

Table 2: Sarcasm detection algorithm

```
Sarcasm models training
Input: dataset
Output: model
    1. sarcasmData ← readDataset()
    2. model ← fitSVMData(sarcasmData)
    3. model ← fitLSTMData(sarcasmData)
    4. return model
```

As we described in table 2, we started by reading the pre-processed dataset. Then we applied a Support Vector Machine algorithm to train our model. For the next step, we made the same for the LSTM model. And finally, we returned our model.

The experiments give us an accuracy of 0.81 for the SVM model and 0.85 for the LSTM model. So, we decided to work with the LSTM algorithm.

Table 3: subjectivity detection algorithm

```
Subjectivity models training
Input: dataset
Output: model
1.  subjectivityData ← readDataset()
2.  model ← fitSVMData(subjectivityData)
3.  model ← fitLSTMData(subjectivityData)
4.  return model
```

As we described in table 3, we started by reading the pre-processed dataset. Then we applied a Support Vector Machine algorithm to train our model. For the next step, we made the same for the LSTM model. And finally, we returned our model.

The experiments give us an accuracy of 0.82 for the SVM model and 0.87 for the LSTM model. So, we decided to work with the LSTM algorithm.

4.6 Sarcasm and Subjectivity Variables

In this area, we will give a brief explanation of the algorithms that we used to train the models for sarcasm and subjectivity detection.

4.6.1 Sarcasm detection with LSTM algorithm

We started by loading our dataset. After that, we applied a pre-processing to make it clean, then we started by defining the algorithm variables. we applied multiple tests and training to our model to sort it with a model having good accuracy.

As we mentioned before, we worked with the Long-short Term Memory approach. We started by sitting max_features to 2000 words. Also, we worked with four layers- Embedding, SpatialDropout1D, LSTM, Dense-, then we split our data, and we took 33% of the data for the test to our model. For the activation function, we used SoftMax. Also, we used Adam optimizer and cross-entropy for the losses. Then we used 25 epochs.

Table 4: Brief explanation of Sarcasm algorithm

```
Sarcasm LSTM Algorithm
1.  data = loadCleanDataWithPanda()
2.  datatoken = tokenazeData(data)
3.  model = createLSTMModel()
4.  model.add(EmbeddingLayer)
5.  model.add(SpatialDropout1D())
6.  model.add(LSTM())
7.  model.add(Dense())
8.  train, test = train_test_split(0.33))
9.  model.fit(train,test,epchos=25)
```

- Embedding: works on transforming data into same-size dense vectors.
- LSTM: used to learn terms dependencies to make it ready for natural language processes.
- Dense: while the LSTM layer's output is not a SoftMax, it's necessary to add the dense layer.

The quality of a model is based on the equation below:

$$Accuracy = \frac{correct\ amount\ of\ guesses}{total\ amount\ of\ guesses} \quad (1)$$

The accuracy of this model we got: 0.85.

4.6.2 Subjectivity detection with LSTM algorithm

As we mentioned in the sarcasm detection algorithm, we also pre-processed and cleaned our subjectivity and objectivity data.
Also, every text with an opinion word is labeled as subjective, whereas the other is labeled as objective.

For subjectivity detection, we worked with the same process as sarcasm by changing the number of epochs to 30.

The accuracy of this model is 0.85.

Due to the limitation of the numbers of data to train our models, we thought we could get more labeled data to achieve good accuracy.

4.7 Long-term Short Memory

In this step, we will talk about the Long-term Short Memory approach and how we implemented it in our research.

After several tests and experiments, we sorted the best parameters to work and train our model.

Before starting working on our algorithm, we pre-processed our IMDB dataset to make it cleaner and applied the same pre-processing we did for the tweets collected data.

At first, we started by reading our dataset using pandas' libraries. After the experiments that we did to test our model, we found that 2500 max_features will be good so that our model will be capable of working with a good number of keywords. Second, we moved to tokenize our dataset using Tokenizer that we got from Keras. After that, we created our model with four layers: Embedding, SpatialDropout1D, LSTM, Dense-. And we set our compiler to Adam as an optimizer. Next, we moved to divide our dataset to train and test the data by putting 33% of the data to be considered for testing purposes. Next, fit the training data to our model, and we set the epochs to 25, with 500 to the batch size. And finally, we evaluated our model to get accuracy.

The accuracy of this model we got: 0.87

4.8 Bidirectional Encoder Representations from Transformers

In this step, we will explain how we worked with the BERT approach to train the model.

As usual, we loaded the cleaned dataset with pandas to ensure the same pre-processing for the dataset and the data collected.

For the BERT approach, we started by using the texts_from_df function. Its rule is to pre-process the text in a way that the model will be capable of understanding. For the parameters of this function, we set it as we can see below in table 5.

Table 5: parameters of texts_from_df function.

texts_from_df
- Train_df = OURTrainData
- Text_column= 'text'
- Label_columns='labels'
- Val_pct=1
- Maxlen=128,
- Preprocess_mode='bert'

For val_pct, we took 10% of the data for testing purposes. Also, we took a maxlen of 128 as recommended, and the mode was BERT, so it will understand that it will be based on BERT processing. As a result, we will get a pre-process, training, testing outputs.

After that, we moved to create our classifier using text_classifier with the parameters shown below in table 6.

Table 6: parameters of text_classifier function

Text_classifier
- Name = 'bert'
- Train_data= (X_train, y_train)
- Preproc= pre-process

After that, we moved to our learner rate, so we started by getting our learner by using the get_learner function, with the parameters shown in the table below.

Table 7: parameters of text_Get_learner function

Get_learner
- Model= OurModel
- Train_data=(X_train,y_train)
- Val_data=(X_test,y_test)
- Batch_size=16

We give a batch size value of 16 because it's recommended to use 116 in batch size with a maxLen of 128. We used a function called lr_find () to get our learner rate. In our case, we got a learner rate equal to 2e-5.

To fit the model, we used a fit_onecylce with the learning rate value and 4 epochs. In the BERT model, it's recommended to use 4 or fewer epochs, usually, 2 epochs will give a good result. Then we saved our model, and we used the get_predictor function to predict our collected data.

5. RESULTS

In this section, we will show the results, a comparison of our algorithm that we used, and how our variables- Sarcasm and Subjectivity detection-, affected the USA elections' results. Also, we will compare our results with other sources.

Before we start, we should mention that how the algorithm worked to give the final results. After applying all the steps that we described before, we got a classification of each text, whether positive or negative. Then we divided our texts into two categories: Trump's keywords and Biden's keywords. If a text is in Trump's keywords and has a positive label, we will add one vote to Trump's, and if it's negative, then the vote will be counted to Biden's. The same thing will happen when a tweet is labeled sarcastic; the vote will counter to the other member.

5.1 SAF Global Results

This area will show our results for the US elections, with LSTM and BERT approaches. Also, we will show the affectation of the variables that we added to SAF "Sentiment Analysis Framework".

Table 8: LSTM and BERT results

	Trump's	Biden's
LSTM	45%	55%
BERT	43%	57%

As we can see in table 8, with LSTM and BERT approaches, Biden's won the USA elections. With a slice different in the percentages of the votes.

That means that Biden's has a chance to win the US elections with 55% than Trump's with 45% in the LSTM approach

A poll was made by Newstatesman that is considered as a British magazine. It predicted that Biden would get 53% of votes in the zero-day, and Trump will get 47% [24].

In the final results, we can see that Biden won with 50.8% of votes and Trump got a 47.5%- until now; we don't have a final result of Georgia State- [25].

Now we are going to present the results with the variables that we added. We should mention that 39% of the data here was categorized as not sarcastic. Also, 37% of the data was categorized as objective with no opinions. So, the table below represents a percentage of estimated results based on the LSTM and BERT with the two variables- Sarcasm detection and Subjectivity Detection-.

Table 9: Framework results with the added variables

LSTM + Sarcasm	44%	56%
BERT + Sarcasm	43%	57%
LSTM + Subjectivity	45%	55%
BERT + Subjectivity	43%	57%
LSTM + Sarcasm + Subjectivity	44%	56%
BERT + Sarcasm + Subjectivity	43%	57%

5.2 SAF Results Based On Months

Our data were collected during three months, August, September, and October.

In the table below, we will compare our LSTM and BERT approaches, with Economist [26]. and CNBC [27], estimated results.

Table 10. SAF results based on months

	LSTM	BERT	Economist	CNBC
August	55% To Biden's	58% To Biden's	54% To Biden's	50% To Biden's
September	55% To Biden's	57% To Biden's	53% To Biden's	51% To Biden's
October	60% To Biden's	62% To Biden's	54% To Biden's	51% To Biden's

5.3 SAF Results Based On States

In this part, we are going to show the map results based on the US states. We must also mention that we have just 8200 tweets divided into all the states for this experiment. Moreover, due to the Twitter API, most of the time, we cannot get the user's location, which is due to the users' privacy.

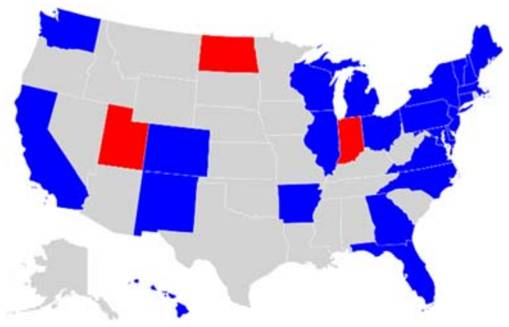

Figure 3: SAF map results

As we can see in Figure 3, the blue color represents Biden's and the red one represents Trump's, and for the grey areas, it means we didn't get any information there.

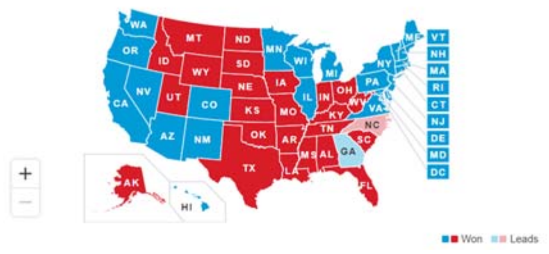

Figure 4: US map elections results

As we can see in Figure 4, we have different results between SAF results and final map results in the center of the map, which is due to the lack of data; as we said before, these are just 8200 tweets divided for all the states.

5. CONCLUSION

In this work, we tried to improve all the techniques and algorithms. We used it before in our other papers. Also, we tried to give a good explanation of each step that we work on it, by giving more references and examples. in order, to help us to ensure quality content for readers who are interested in sentiment analysis and machine learning. Also, the limitations that we have in this experiment, was time-consuming to train the data and prepared our models. Also, the same things happened when we need to predict the sentiment of our tweets, due to the big amount of collected data. But we moved to use Azure Machine learning tools, to speed our process.
Our proposed SAF- Sentiment Analysis Framework- integrates many methods and approaches to analyze sentiment on data extracted from social media. The case study we implemented shows that the results are accurate and on the point. Also, we can mention that there are many ideas that we can execute to enhance our IM sentiment analysis framework performance and to get even better results with high accuracy. The Sarcasm detection and Subjectivity detection approaches in the framework's sentiment analysis section could be improved by finding or creating a huge dataset. So, the model can have more data to work with it. We can also use some other approaches, such as combining lexicon-based methods with machine learning to have better accuracy in Sarcasm and Subjectivity detection cases. For the sentiment analysis of a tweet, and after we applying the LSTM and BERT approaches, we noticed that it is a good opportunity to improve a sentiment analysis field and increase the accuracy of the results.

In our work, we started by defining the ontologies to get just the important and related data to our subject. Then we moved to collect data from Twitter based on ontologies that we created. After that, we used an NLTK approach to clean our data to be ready for our Framework's next step. Then we applied subjectivity and sarcasm detection to all our cleaned tweets that we extracted from social media. After creating the LSTM and BERT models, we tried to predict the sentiment of the tweets based on these models to sort with a labeled tweet, either positive or negative. And finally, we moved to the visualization part containing some calculation script to present our results in different styles to make it easy for the viewer to analyze and compare the results.

For the next works, we aim to implement a new architecture of Big data, and make our visualization part in real-time. We aim to work deeply with BERT approaches and test other LSTM variables to further understand these approaches and to increase their accuracy. Also, we aim to use ontologies and find other uses to ensure that our collected data is fitting our needs. We aim to find a solution to increase the subjectivity and sarcasm detection accuracy because we think these two approaches are necessary for the sentiment analysis field.

ACKNOWLEDGEMENT:

This research was financially supported by CNRST, as known as the National Center for Scientific and Technical Research of Morocco. The Authors wish to give their sincere thanks to DR. Jamila EL ALAMI Director of CNRST for valuable cooperation.

REFERENCES:

[1] Apoorv Agarwal Boyi Xie Ilia Vovsha Owen Rambow Rebecca Passonneau, Sentiment Analysis of Twitter Data.
[2] Sarlan, Aliza & Nadam, Chayanit & Basri, Shuib. (2014). Twitter sentiment analysis. 212-216. 10.1109/ICIMU.2014.7066632.
[3] Yogesh Garg and Niladri Chatterjee, Sentiment Analysis of Twitter Feeds,

https://doi.org/10.1007/978-3-319-13820-6_3, print ISBN 978-3-319-13819-0.

[4] Rana Abaalkhail, Benjamin Guthier. and Abdulmotaleb El Saddik, Human Affective States Ontology for Sentiment Analysis, 1570-0844/0-1900/.

[5] Ting-Wei Liu, Xiong Luo, Long Wang, An LSTM Approach to Short Text Sentiment Classification with Word Embeddings, The 2018 Conference on Computational Linguistics and Speech Processing ROCLING 2018, pp. 214-223 ©The Association for Computational Linguistics and Chinese Language Processing.

[6] Shivaji Alaparthi, Manit Mishra, Bidirectional Encoder Representations from Transformers (BERT): A sentiment analysis odyssey, https://arxiv.org/abs/2007.01127.

[7] Manish Munikar, Sushil Shakya and Aakash Shrestha, Fine-grained Sentiment Classification using BERT, arXiv:1910.03474v1 [cs.CL] 4 Oct 2019.

[8] M. Schuster and K. Nakajima, "Japanese and korean voice search," in 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2012, pp. 5149–5152

[9] Mickel Hoang, Oskar Alija Bihorac, Jacobo Rouces. Aspect-Based Sentiment Analysis Using BERT. Nodalida 2019.

[10] Moudhich Ihab, Loukili Soumaya, and Fennan Abdelhadi, Sentiment Analysis. A comparative of machine learning and fuzzy logic in the study case of Brexit sentiment on social media, DOI: 10.1145/3368756.3369090.

[11] Efstratios Kontopoulos, Christos Berberidis, Theologos Dergiades, Nick Bassiliades. Ontology-based Sentiment Analysis of Twitter Posts. 40. 4065-4074. 10.1016/j.eswa.2013.01.001.

[12] Merfat M. Altawaier and Sabrina Tiun, Comparison of Machine Learning Approaches on Arabic Twitter Sentiment Analysis, Vol.6 (2016) No. 6, ISSN: 2088-5334.

[13] Efthymios Kouloumpis, Theresa Wilson and Johanna Moore, Twitter Sentiment Analysis: The Good the Bad and the OMGs!, Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media

[14] Hamid Bagheri, Md Johirul Islam, Sentiment analysis of Twitter data (2017).

[15] Luciano Barbosa and Junlan Feng, Robust Sentiment Detection on Twitter from Biased and Noisy Data, Coling 2010: Poster Volume, pages 36–44, Beijing, August 2010.

[16] Yessi Yunitasari, Aina Musdholifah, Anny Kartika Sari, Sarcasm Detection for Sentiment Analysis in Indonesian Tweets, DOI: https://doi.org/10.22146/ijccs.41136.

[17] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schutze (2017). Comparative Study of CNN and RNN for Natural Language Processing. arXiv:1702.01923v1 [cs.CL] 7 Feb 2017

[18] Jin Wang, Liang-Chih Yu, K. Robert Lai, and Xuejie Zhang, Dimensional Sentiment Analysis Using a Regional CNN-LSTM Model, Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 225–230, Berlin, Germany, August 7-12, 2016. c 2016 Association for Computational Linguistics.

[19] Yukun Ma, Haiyun Peng, Erik Cambria, Targeted Aspect-Based Sentiment Analysis via Embedding Commonsense Knowledge into an Attentive LSTM, Conference: AAAI At New Orleans.

[20] sporia, cambria, devamanyu, and Prateek, A Deeper Look into Sarcastic Tweets Using Deep Convolutional Neural Networks, arXiv:1610.08815v2 [cs.CL] 27 Jul 2017.

[21] Tomas Pt' acek, Ivan Habernal, and Jun Hong. 2014. Sarcasm detection on Czech and English' Twitter. In COLING, pages 213–223.

[22] I.V., Shravan. (2016). Sentiment Analysis in Python using NLTK. OSFY – Opensource

[23] Z. Jianqiang and G. Xiaolin, "Comparison Research on Text Pre-processing Methods on Twitter Sentiment Analysis," in IEEE Access, vol. 5, pp. 2870-2879, 2017, DOI: 10.1109/ACCESS.2017.2672677.

[24] NewStatesMan, https://www.newstatesman.com/international/2020/11/us-presidential-election-2020-donald-trump-or-joe-biden-leading-our-poll.

[25] Theguardian, https://www.theguardian.com/us-news/ng-interactive/2020/nov/12/us-election-results-2020-joe-biden-donald-trump-presidential-electoral-college-votes.

[26] Economist, https://projects.economist.com/us-2020-forecast/president.

[27] CNBC results, https://www.bbc.com/news/election-us-2020-54094119.