Social media based sentiment analysis: Natural Language Processing

Giray Kılıç
Computer Engineering
İstanbul Kültür University
İstanbul, Turkey
1600002297@stu.iku.edu.tr

Oğuz Kaliç
Computer Engineering
İstanbul Kültür University
İstanbul, Turkey
1600004825@stu.iku.edu.tr

Cengiz Diker
Computer Engineering
İstanbul Kültür University
İstanbul, Turkey
1600003909@stu.iku.edu.tr

I. Dataset

A. Text Scraping

Sentiment analysis needs text data for educational sentiment analysis on Twitter. Data from Twitter had to be scrapped. Two libraries could be used for this. The first is the twitter API [1] that twitter has created for python. In order to use this library, a statistical research had to be done for a commercial purposes, but our study only covers academic research. It is not possible to use this library for this. Another alternative was to use an open source library called ”twint” [2]. Therefore, this was chosen as the appropriate library. The ”pandas” [3] library was chosen for data extraction and also for data analysis.

B. Language Selection

After the libraries were selected to extract basic data, it was considered in which language to extract the data. It was decided Turkish, but even if there was a library to do Turkish sentiment analysis, their documentation was very insufficient. Therefore English was selected. Textblob library [4] was chosen as the library where sentiment analysis for English.

C. Data and Features

A total of one hundred thousand tweets were taken. The reason it cannot be more is because there is a certain time limit for data request. After a certain time, the connection timeout and stops the scraping process. This causes loss of all scraped data. Ten Twitter tags were used within a hundred thousand data. Starting date was set as April 17, 2020. There totally thirty nine features in data nineteen of them was removed Tab. I. Foreign Language or empty tweets filtered out by language column then removed duplicate tweets about seventy thousand tweet remained. After preprocessing subjectivity and polarity column have been added.

TABLE I
COLUMNS AFTER FEATURE SELECTION

Features	Description
id	Tweet id
conversation_id	Conversation id
created_at	Creation Geolocation
date	Timestamp of tweet
tweet	Tweet (message)
hashtags	Filtered hashtags from tweet
cashtags	Filtered cashtags from tweet
user_id	Id of user
user_id_str	Id of user string type
username	Username of tweet
name	Real Name
urls	List of URLS in tweet
photos	List of photo links
video	List of video links
retweet	Replies
nlikes	Number of likes
nreplies	Number of replies
nretweets	Number of retweets
reply_to	Tweet who replied for
polarity	Polarity score from Textblob
subjectivity	Subjectivity score from Textblob

II. Goals

A. Detecting Three Different Emotions

We tried to detect three different emotions in this project. First, we try to detect positive tweets. Second, we detected neutral tweets about the distance learning. Third, we tried to detect negative tweets about distance learning. In the end the distribution about those three emotions were not equal. So, we have to somehow eliminate some tweets in the process.

B. Personal Comments

The goal of this project is enhance our Artificial Intelligence knowledge. How the distance learning effects our education system. Our project aim is to find about these twitter comments.

III. Data Preprocessing

A. Deletion of non-Ascii Characters

Starting for a healthy preprocess, firstly need to remove characters outside of ascii characters. Since these characters will disrupt your emotional analysis and embedding vector.

B. Starting Preprocessing

Links, mentions, hashtags and digits removed by regular expressions. Then punctionations removed. All words are lowered and tokenized [5].

C. Bigrams and N-grams

The use of this stage is uncertain. Some words can be found in groups so gensim's automatic phraser is used. However, it may not give a good result in every case. In manual, it can be put into phrases such as bigrams and trigrams, but this would take a lot of time.

D. Lemmatization

Tokenized words are firstly lemmatized by noun, adjective and verb. If the token is not lemmatized then it will be spellchecked. In case of spell is wrong it will be looked if token is in abbrevation list [6]. If not abbrevation then word added not found word index and it will be removed from token list.

E. Removing Short Tokens

In case of some short words pretends like stopwords. Therefore two or less character tokens will be removed.

F. Calculating Polarity and Subjectivity Scores

When all preprocess is done sentiment analysis starts and polarity and subjectivity scores will be calculated. However only polarity will be used. Subjectivity maybe used in future. Finnally polarity label starts to be created . This tag is divided three according to goal. If polarity is zero then it is labeled by one. If smaller it is negative it is labeled zero and if bigger then zero it is labeld two. These options could changed according to results and support.

IV. Method

A. LSTM Layer

In our model we first used an LSTM layer with 128 outputs with recurrent dropout 0.5 value. An LSTM layer learns long-term dependencies between time steps in time series and sequence data.The layer performs additive interactions, which can help improve gradient flow over long sequences during training.

B. Gates

In LSTM we will have 3 gates:

- Input Gate.
- Forget Gate.
- Output Gate.

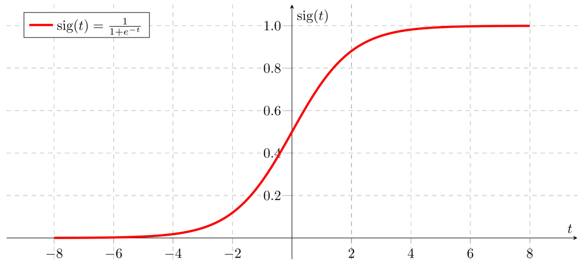

Fig. 1. Gates in LSTM are the sigmoid activation functions i.e they output a value between 0 or 1 and in most of the cases it is either 0 or 1.

We use sigmoid function [7, Fig. 1] for gates because, we want a gate to give only positive values and should be able to give us a clear cut answer whether, we need to keep a particular feature or we need to discard that feature.

- "0" means the gates are blocking everything.
- "1" means gates are allowing everything to pass through it.

The equations for the gates in LSTM [7, Eq. 1 and 2]

$$\begin{aligned} i_t &= \sigma(w_i[h_{t-1}, x_t] + b_i) \\ f_t &= \sigma(w_f[h_{t-1}, x_t] + b_f) \\ o_t &= \sigma(w_t[h_{t-1}, x_t] + b_t) \end{aligned} \tag{1}$$

$$\begin{aligned} i_t &\rightarrow represents\ input\ gate. \\ f_t &\rightarrow represents\ forget\ gate. \\ o_t &\rightarrow represents\ output\ gate. \\ \sigma &\rightarrow represents\ sigmoid\ function. \\ w_x &\rightarrow weight\ for\ the\ respactive\ gate(x)\ neurons. \\ h_{t-1} &\rightarrow output\ of\ the\ previous \\ &\quad lstm\ block(attimestamp\ t-1). \\ x_t &\rightarrow input\ at\ current\ timestamp. \\ b_x &\rightarrow biases\ for\ the\ respective\ gates(x). \end{aligned} \tag{2}$$

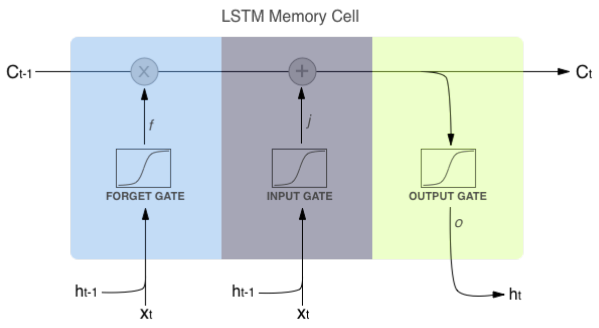

Fig. 2. First equation is for Input Gate.It tells us new information going to store in the cell state. Second is for the forget gate which tells the information to throw away from the cell state.Third one is for the output gate which is used to provide the activation to the final output of the lstm block at timestamp 't'. [7]

C. Dense Layer and Softmax

Followed by a dense layer with 3 outputs and softmax activation function.(0=negative, 1=neutral, 2=positive) Dense layer is the regular deeply connected neural network layer. It is most common and frequently used layer. Dense layer does the below operation on the input and return the output.

Softmax is often used as the activation for the last layer of a classification network because the result could be interpreted as a probability distribution [8].

Furthermore, it tells us that a dense layer is the implementation of the equation

$$output = activation(dot(input, kernel) + bias)$$

This means that we are taking the dot product between our input tensor and whatever the weight kernel matrix is featured in our dense layer [9].

D. Dropout

The reason we used dropout in this project, when using dropout regularization, it is possible to use larger networks with less risk of overfitting. In fact, a large network (more nodes per layer) may be required as dropout will probabilistically reduce the capacity of the network [10].

E. Word Embeddings with word2vec

In addition, We used Word Embeddings with word2vec in the model for the text data on the first input layer.

Word2Vec [[11], Fig. 3] is an unsupervised (no labels) and prediction-based model that tries to express words in vector space.

- model architectures
- skip-gram (SG)
- continuous bag of words (CBOW)

Objective is to train an embedding matrix, where the number of rows is the size of the vocabulary and the number of columns is the number of dimensions used to represent words in the vocabulary [12].

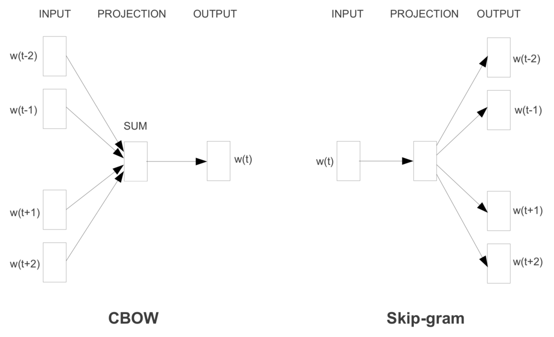

Fig. 3. Word2Vec architecture

V. Results

$$\begin{bmatrix} 1706 & 408 & 341 \\ 187 & 8082 & 306 \\ 287 & 678 & 11197 \end{bmatrix} \tag{3}$$

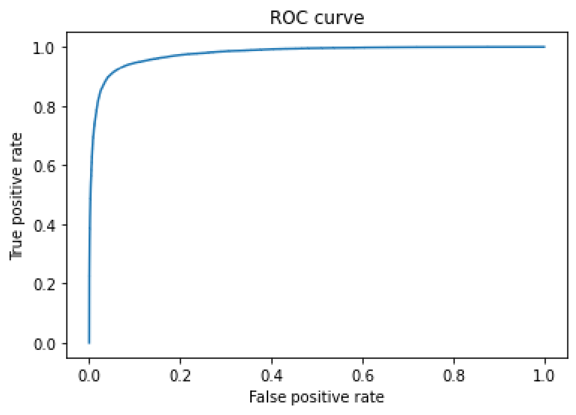

Fig. 4. ROC curve AUC: 97.789

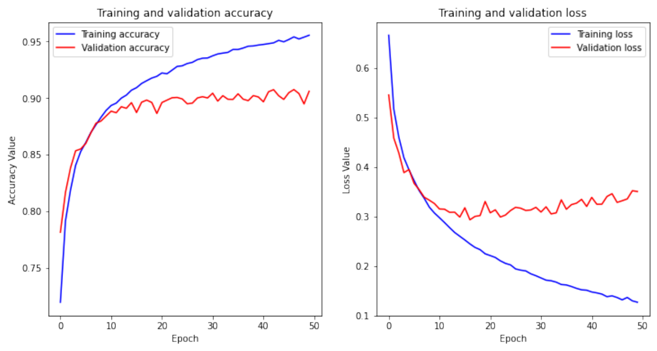

Fig. 5. Accuracy Rates and Error Rates red validation blue test

TABLE II
RESULTS FROM TEST DATA

	Precision	Recall	F1-Score	Support
0	0,78	0,69	0,74	2455
1	0,88	0,94	0,91	8575
2	0,95	0,92	0,93	12162
accuracy			0,9	23192
macro average	0,87	0,85	0,86	23192
weighted average	0,9	0,9	0,9	23192

VI. Conclusions

Negative tweets should be incremented in this system. In this project the difference between negative, neutral and positive tweets are so vast. Therefore, recall and precision rates are so different. If the project has more negative tweets about distance learning we would have more accuracity about distance learning.

After the twentieth epoch, the project start to begin an overfitting. Models negative numbers were so low compare the the neutral and positive numbers. Therefore, the results are we getting is not efficient in this project.

Label polarity numerical values could be changed in this project.

References

[1] "Twitter api documentation — twitter developer." [Online]. Available: https://developer.twitter.com/en/docs/twitter-api

[2] "twint." [Online]. Available: https://github.com/twintproject/twint

[3] "About pandas." [Online]. Available: https://pandas.pydata.org/about/

[4] S. Loria, "Simplified text processing." [Online]. Available: https://textblob.readthedocs.io/en/dev/

[5] B. Hasdemir, "Bhasfe/distance_learning." [Online]. Available: https://github.com/Bhasfe/distance_learning

[6] "Netlingo: Every texting acronym & online abbreviation you'll ever need to know." [Online]. Available: https://www.netlingo.com/acronyms.php

[7] D. Thakur, "Lstm and its equations," Jul 2018. [Online]. Available: https://medium.com/@divyanshu132/lstm-and-its-equations-5ee9246d04af

[8] J. Brownlee, "Softmax activation function with python," Jun 2020. [Online]. Available: https://machinelearningmastery.com/softmax-activation-function-with-python/

[9] H. Heidenreich, "Understanding keras - dense layers," Jan 2019. [Online]. Available: https://medium.com/@hunterheidenreich/understanding-keras-dense-layers-2abadff9b990#: :text=Furthermore, it tells us that,featured in our dense layer.

[10] J. Brownlee, "A gentle introduction to dropout for regularizing deep neural networks," Aug 2019. [Online]. Available: https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/

[11] C. Shuai, "word2vec," Jul 2019. [Online]. Available: https://nocater.github.io/2018/12/29/word2vec/

[12] M. Buyukkinaci, "Word2vec nedir ? (türkçe)," Jan 2018. [Online]. Available: https://medium.com/@muhammedbuyukkinaci/word2vec-nedir-türkçe-f0cfab20d3ae