Sentiment and Emotion Recognition from Texts Using Deep Neural Network

Abdullah Arafat Miah
Electrical, Computer, and Biomedical Engineering
University of Rhode Island
Kingston, RI
abdullaharafat.miah@uri.edu

***Abstract*—Sentiments and emotion recognition from texts are highly valued in today's world because of their numerous applications in market research, social filtering, improved healthcare systems, etc. In this project, we have applied a Long Short-Term Memory (LSTM) based deep neural network with an additional attention layer for classifying sentiments and emotions from texts. We have done several standard Natural Language Processing (NLP) techniques like data cleaning, word-level tokenization, and word-to-vector conversion to get improved classifying results. The performance of the model was evaluated using four different evaluation metrics on three different data sets in two languages. The codes were written in Python language mostly using the PyTorch library for building and training the deep neural network. The experiments were run using a laptop having RTX 3050 GPU.**

***Index Terms*—Natural Language Processing, Sentiment and Emotion Analysis, Long Short-Term Memory (LSTM), Attention**

I. Introduction

Sentiment refers to the polarity of the emotions. Usually it can be divided into three types. They are: Positive, Negative, and Neutral. Emotions have many shaded. Most commons are happiness, anger, sadness, fear, surprise, amusement, joy, etc. A person's text can have these kinds of sentiments and emotions. People usually express their emotions through tweets, Facebook status. In today's world, where social media is so prevalent for expressing one's opinion, it becomes more important to classify one's sentiment and emotions from texts for variety of reasons. It is useful for evaluating customer insights, brand reputation management, market research, social listening, social filtering, improving healthcare and well-being etc. As sentences can be represented as time dependent sequence of words Recurrent Neural Network (RNN) [2] [3] is the best suited network for these kind of tasks. In this project, we used an special type of RNN called long short term memory (LSTM) network [1] with an extra attention layer to classify sentiments and emotions from texts. The attention layer is responsible for focusing on a specific word from the sentence for classification. We did several standard data pre-processing techniques like word level tokenization, word to feature vector conversion etc., before feeding the sentences to our neural network. The performance of the network is evaluated using three different data sets in two languages and four evaluation metrics.

II. Data Pre-Processing

A. Data Cleaning

Every data set goes through he data pre-prossessing part and the first step is the data cleaning. The purpose of the data cleaning is to get rid of unwanted words or characters of a sentence that is essential for making the sentence structure correct but don't actually convey any meanings. With this cleaning the neural networks don't have to deal with unwanted parts of a sentence. Data cleaning can be divided into four steps. They are:
01. Replacing newline and carriage return with space, and convert to lowercase. This symbols is not needed for conveying any meaning of the sentence.
02. Removing links and mentions.
03. Removing non-ASCII characters.
04. Removing stop words. Stop words means: "the," "is," "and," "in," etc.

Suppose there is a sentence in a dataset: "*What does it take to delight stakeholders with the online experience? This whitepaper on @Dell Technologies APEX shows how IT transformation paves a path to the post-hybrid #cloud era. Learn how the team at accel bi corporation can get you started. https://t.co/sBaKBIJQn1 https://t.co/ujDzMcCJLe*". After doing the data cleaning operation, the sentence will be: "*take delight stakeholders online experience whitepaper technologies apex shows transformation paves path posthybrid cloud era learn team accel bi corporation get started* ".

B. Tokenization

Tokenization means braking down a sentence into fundamental units named as tokens. There are several common methods for tokenizations. They are:
01. Word level tokenization: Here the sentences are splitted into individual words. Each words are regarded as tokens.
02. Sentence level tokenization: In sentence level tokenization, the texts are spliited into sentences. This methods is fruitful when there are large passages in data sample in a dataset.
03 Subword level tokenization: Subword level tokenization breaks words into smaller subwords. In this type of tokenization there is always a pre defined vocabulary.
In our project, we utilize the word level tokenization. For a data sampple: "*take delight stakeholders online experience whitepaper technologies apex shows transformation paves path posthybrid cloud era learn team accel bi corporation get started*" will be converted to "*['take', 'delight', 'stakeholders', 'online', 'experience', 'whitepaper', 'technologies', 'apex', 'shows', 'transformation', 'paves', 'path', 'posthybrid', 'cloud', 'era', 'learn', 'team', 'accel', 'bi', 'corporation', 'get', 'started']*" after word level tokenization. After splitting each samples we gathered all the words in a data set and make our vocabulary dictionary. We denote each word of the dictionary a token ID. Then previous tokenized can be denoted as: "*[0, 0, 0, 0, 0, 0, 0, 0, 59, 2591, 2325, 271, 55, 1375, 106, 477, 415, 837, 2592, 1861, 2850, 160, 1587, 81, 50, 4712, 4334, 2058, 6, 252])*". The purpose of denoting token id is that computers work well with numerical numbers. We defined a fixed length for each samples. We add 0 in the beginning of the each samples if they are shorter than the length. If they are larger than the length then we discarded the last words. This process is called padding.

C. Word2Vec

By the word2vec we can convert a word into a vector of high dimensional space. The purpose of converting a vector to a high dimensional space is that distributed representations helps words to be expressed based on its context in the corpus of texts. This representation is also known as word embedding. In word2vec, shallow neural network is trained to predict a word within its context. There are two ways of learning. One is continues bag of words (CBOW), where the network is trained the target word within its surrounding context. Another is Skip-gram, where the network is trained to predict the context word given the surrounding words. After training, the dense layer before the classification layer represents the vector of the target word. The dimension of the vector is user defined. We used 100 dimensional vectors to represent each words. More details of the word2vec method is given at [4].

D. Data Distribution:

Every data sets is divided into three parts: training data set (80%), testing data set(10%), validation data set(10%). All the results that will be shown in the later parts are based on testing part of each data sets.

III. Network Explanation:

A. Typical recurrent neural network

Recurrent neural network is mainly used for sequential data. In standard neural network input data is static and the temporal relationship between the data is not considered. Unlike traditional neural network there is a importance of prior input to present input and output. There is a flow of information from one hidden layer to another hidden layer in temporal direction. Recurrent neural networks share the same weight parameter within each layer of the network. Each node in the hidden usually have tanh as the activation function. Recurrent Neural Network is sensitive to vanishing and exploding

gradients. It does not work well with long sentences. So it may affect the performance of our model. So we used another updated version of the RNN, Long Term Short Memory (LSTM) [1] for this task.

B. Long Term Short Memory (LSTM)

In RNN there is a flow of information from one hidden state to another hidden state. But for LSTM there is an additional flow of information which is called memory flow. This additional flow determines which part of the sequence should be remembered and which part should not be. Traditional LSTM cell has three gates. Forget gates, which eliminate the unnecessary information, Input gate which add or update new information and output gate where we get the sequential output. Hidden state is the output from the last LSTM cell which contains the the information of whole sequence.

C. Proposed Deep Neural Network

In the 1 a visual description of our proposed deep neural network is given.

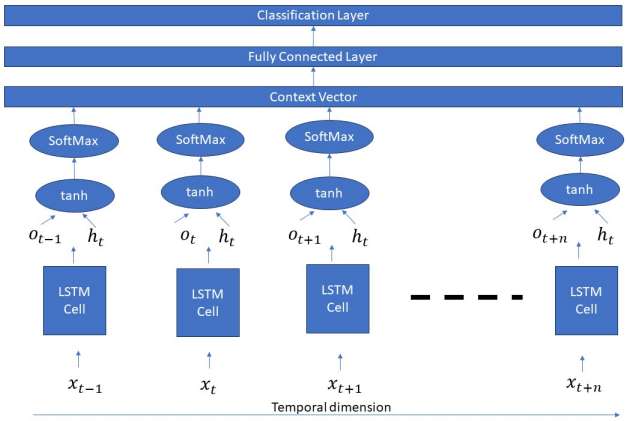

Fig. 1. Proposed Deep Neural Network.

Here, the x_{t-1}, x_t, x_{t+1}, x_{t+n} denote the input word embeddings of the input sentences with a length of n. They go through one-directional LSTM cells according to the temporal direction. o_{t-1}, o_t, o_{t+1}, o_{t+n} denote the output of the LSTM cells which is usually the output of the output gates. h_t is the hidden state. Each cell's output and the hidden state go through the $tanh$ activation function and then SoftMax is applied to display the output in probabilistic distribution with a range from 0 to 1. The outputs are attention weights and the weights consist the context vector. This layers are known as attention layers. Context vector has the information of all the sequence and during training through this vecoto the network will learn which part of the input sentence should emphasize more. The context vector layer is than connected with a fully connected layer and finally it through log-softmax function it will connect to a classification layer. The classification layer will classify the input sentence into a sentiment or emotion.

D. Model Parameters

No of feature vectors in word embedding layer is 100. The Loss function used during training is Cross-entropy. During backpropagation for updating weights the optimizer used is Adam [5]. The batch size wash 16. Dropout percentage 50%. And number of epochs is equal to 10.

IV. Experimental Results

A. Benchmark Data sets

Dell Tweets 2022 [6] is a publicly available data set which contains 25000 tweets collected from the social media platform twitter. This twitter contains all the tweets which mention the name 'Dell' during the Jan 01 – Sept 30, 2022. Along with the tweets it contains date, timestamp, username, and tweet Id. The tweets are labelled based on sentiments and emotions. There are three sentiments for each tweet: positive, negative, and neutral. And emotions are joy, love, optimism, pessimism, trust, surprise, anticipation, sadness, anger, disgust and fear. Sentiment Analysis in Text [7] is another data set used which contains more data than the previous data. It contains 40000 tweets, and it has been labeled through 13 emotions. The dataset has three columns. They are tweet-id, sentiment, and content. As the dataset is larger it will be challenging to build a model based on this data set. Previous two data sets are based on English language. We wanted to train and test out models based on another language. That's why we choose the Turkish tweets data set [8]. This data set contains 5 emotions: anger, happy, distinguish, surprise and fear. It contains 4000 tweets, 800 tweets for each emotion.

B. Programming Language Used

To implement our work we mainly relied on Python with a machine learning framework named PyTorch. Also, we used the Panda library through which its easier to do the data pre-processing steps. We also used Scikit Learn library and Matplotlib library for different tasks for completing the project.

C. Results

Four parameters: accuracy, precision, recall, and F1 score was used for performance evaluation. Accuracy shows the percentage of correctly classified data for a classifier. It is the ratio of the correctly classified data to the total amount of data. Sometimes accuracy alone can't be a good measure. Precision gives us the percentage of correctly classified positive data over all the true (TP) and false (FP) positive data. Recall gives us the percentage of correctly classified positive data over true positive (TP) and false negative (FN) data. F1 score combines both.

2 - 5 contain the classification results and confusion matrix for all the classes in dell tweet - 2022 data set's sentiment and emotion classification. 6 - 7 contain the classification results and confusion matrix for all the classes in turkish tweet dataset's emotion classification. 8 - 9 contain the classification results and confusion matrix for all the classes in Sentiment Analysis in Text data set's emotion classification.

	precision	recall	f1-score
negative	0.76	0.88	0.82
neutral	0.66	0.52	0.58
positive	0.73	0.72	0.72
accuracy			0.73
macro avg	0.72	0.71	0.71
weighted avg	0.72	0.73	0.72

Fig. 2. Dell Tweets 2022 data set (sentiment) classification report.

V. Discussion & Conclusion

In this project, we proposed a deep neural network consisting of LSTM cells and attention layer to recongnize sentiment and emotions from texts. When there are large number of samples in each classification group, the network works well. When there is a balanced data set with notable number

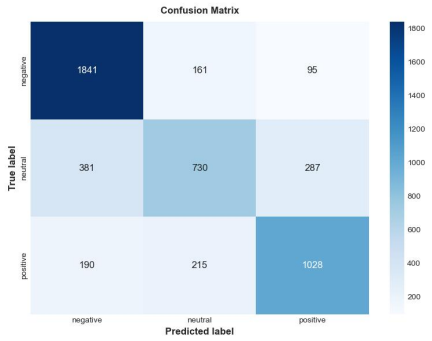

Fig. 3. Dell Tweets 2022 data set (sentiment) confusion matrix.

	precision	recall	f1-score
anticipation	0.52	0.59	0.55
joy	0.68	0.69	0.68
anger	0.74	0.75	0.75
sadness	0.33	0.27	0.29
fear	0.50	0.34	0.41
optimism	0.36	0.38	0.37
disgust	0.34	0.27	0.30
surprise	0.00	0.00	0.00
accuracy			0.59
macro avg	0.43	0.41	0.42
weighted avg	0.58	0.59	0.59

Fig. 4. Dell Tweets 2022 data set (emotion) classification report.

of samples in each group, the model works best (Turkish tweet data set). The model does not work well with very large and imbalanced data sets. But if a group has substantial samples, it can predict the group correctly most of the time. This problem can be fixed by using multi-layer attention mechanism. Which is used in modern transformer-based

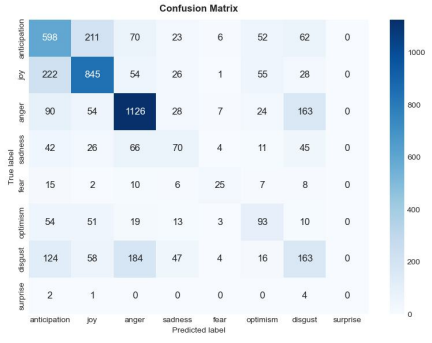

Fig. 5. Dell Tweets 2022 data set (emotion) confusion matrix.

	precision	recall	f1-score
kızgın	0.92	0.91	0.91
korku	0.93	0.91	0.92
mutlu	0.89	0.89	0.89
surpriz	0.88	0.88	0.88
üzgün	0.83	0.86	0.85
accuracy			0.89
macro avg	0.89	0.89	0.89
weighted avg	0.89	0.89	0.89

Fig. 6. Turkish tweet data set classification report.

Fig. 7. Turkish tweet data set confusion matrix.

	precision	recall	f1-score
empty	0.04	0.03	0.03
sadness	0.25	0.31	0.28
enthusiasm	0.01	0.01	0.01
neutral	0.32	0.44	0.37
worry	0.33	0.29	0.31
surprise	0.08	0.06	0.07
love	0.35	0.31	0.33
fun	0.09	0.06	0.07
hate	0.26	0.15	0.19
happiness	0.27	0.29	0.28
boredom	0.00	0.00	0.00
relief	0.06	0.02	0.03
anger	0.00	0.00	0.00
accuracy			0.28
macro avg	0.16	0.15	0.15
weighted avg	0.26	0.28	0.27

Fig. 8. Sentiment Analysis in Text data set classification report.

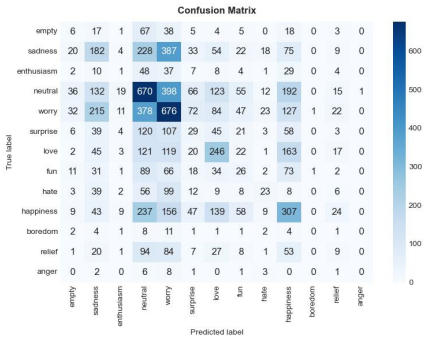

Fig. 9. Sentiment Analysis in Text data set confusion matrix.

architectures.

REFERENCES

[1] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.

[2] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors." nature 323.6088 (1986): 533-536.

[3] Jordan, Michael I. "Serial order: A parallel distributed processing approach." Advances in psychology. Vol. 121. North-Holland, 1997. 471-495.

[4] Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013).

[5] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

[6] Kumar, A. (2022, December 9). Dell Tweets 2022. Kaggle. https://www.kaggle.com/datasets/ankitkumar2635/dell-tweets-2022

[7] Sentiment analysis in text - dataset by Crowdflower. data.world. (2016, November 21). https://data.world/crowdflower/sentiment-analysis-in-text

[8] Guven, A. (2021, April 9). Turkish tweets dataset. Kaggle. https://www.kaggle.com/datasets/anil1055/turkish-tweet-dataset?select=TurkishTweets.xlsx