ISyE 6740 – Spring 2022
Final Report

Team Member Names: Divij Pherwani (903079079)

Team Name: NLP Analyst

Project Title: Applying Deep Learning for Natural Language Processing

Problem Statement

There is a massive amount of data generated every day. For example, over 305 billion emails and 500 million tweets were generated every day in 2020. Structured data are clearly defined and searchable types of data, while unstructured data are usually stored in their native format. Structured data are quantitative, while unstructured data are qualitative. Structured data are often stored in data warehouses, while unstructured data are stored in data lakes. The amount of unstructured data is much larger than that of structured data. Unstructured data makes up a whopping 80% or more of all enterprise data, and the percentage keeps growing. This means that companies not taking unstructured data into account are missing out on a lot of valuable business intelligence.

Social media such as Twitter are a good source to gather public opinion or sentiment on any topic of scale. People such as Elon Musk use it to promote their company; voice their opinion on policies, climate, and/or AI; even use it to indirectly manipulate the cryptocurrency market (the dogecoin incident). Recently, Elon Musk decided to buy the very platform he uses actively in the name of free speech for 43 billion dollars. Therefore, it was interesting to investigate how the people on the platform feel about him i.e., positive, negative, or neutral.

Introduction

Natural Language Processing or NLP is a branch of AI that deals with the interaction between computers and humans using the natural language (English) whereas sentiment analysis is a text mining technique that uses ML and NLP to analyze text for the sentiment (positive, negative, and neutral). The paper will discuss the methods used to extra data from Twitter, interesting insights gathered, techniques used to label the data, and the classification methods used to categorize data based on sentiments. The program was built in Python programming language.

Public opinion and sentiment could be useful information in cases such as election campaigns or polls. The way people express their emotions could be an indicator of their liking of the person. The changes that happen over time could be mapped to changes in the sentiments; the most recent example of this would be the Johnny Depp - Amber Heard Defamation Trial. Public sentiment also has a slight correlation with the stock prices. All these examples prove the importance of investigation sentiments. The ultimate objective of NLP is to read, decipher, understand, and make sense of human languages in a valuable manner.

Sentiment Analysis in the text is not a new task but the differentiating factor of this paper is the integration with Twitter API, creating labels for analysis, and the possibility to do it for any hashtag in the same setup with a simple modification of query. This made the research unique when compared to a standard data competition online or predefined dataset.

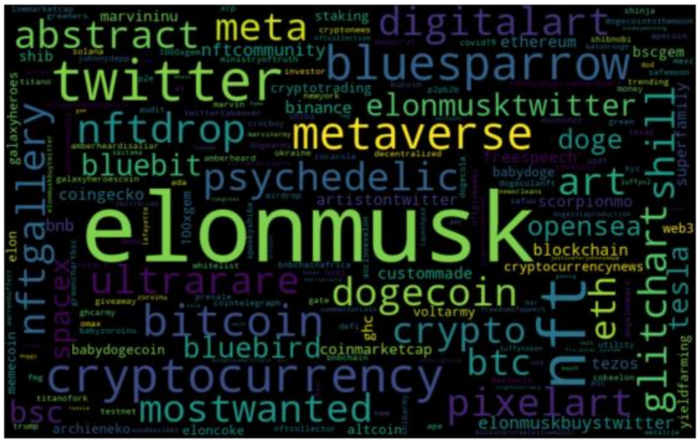
Data Source

In this project, I mined the data from Twitter using tweepy which is an open-source library useful for accessing the tweet data. The data of interest on Twitter is Elon Musk or tweets with #elonmusk and the tweet should be an original (not a retweet). The query is customizable and can work for any given hashtag. For this project, 50k recent tweets were fetched. The API gathers the specified number of tweets from the time model was run in the order of recency. The gathered data is later stored in a CSV file for further analysis and building classification models. Tweepy is very customizable, and the data can be scrapped ad-hoc as per the specified requirements.

The dataset created includes the tweet text (full_text), the user id (id), tweet creation date (date), tweet source (source), the number of users who liked the tweet (likes), and the number of times the tweet was retweeted (retweets). There are many attributes to choose from while extraction such as the location, hashtags, username, etc. but for this project, I decided to fetch only the earlier mentioned attributes. It was a risky decision to do so as it is possible to obtain labeled and clean data from Twitter on Kaggle but linking to API was part of exploration and learning. Also in the real world, it is not always possible to have predefined labels.

Methodology

The saved data in CSV from Twitter will be used as input. The interface will receive input data frames in the form of free text. The tweet is the only attribute I care about as it is where the text is present for analysis. The data will be preprocessed using NLP techniques such as Tokenization, Stemming, stop word removal, hashtags/user mention/ HTTP link processing. Data cleaning is the most important step here because garbage in could lead to garbage out. For exploratory purposes, the world cloud of hashtags in the tweets with #elonmusk is presented below,



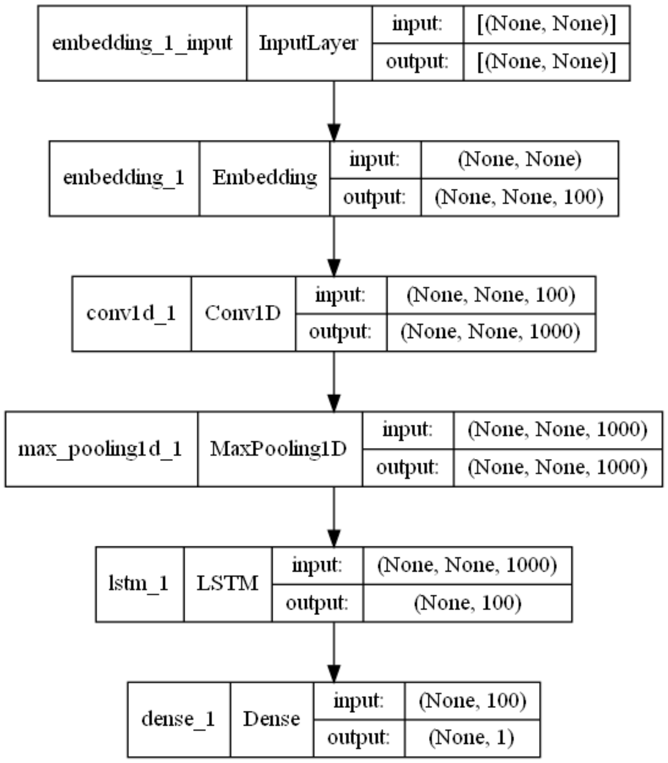
Elon Musk seems popular with crypto and NFT

Before we build classification models, there is a need to label the unlabeled data. It is challenging to do them manually. Therefore, I decided to use two libraries as benchmarks. Textblob and NLTK libraries have built-in methods such as Textblob Sentiment Polarity and Sentiment Intensity Analyzer (SIA) respectively. TextBlob returns the polarity of a sentence that lies between [-1,1]. -1 defines a negative sentiment, 1 defines a positive sentiment, and 0 defines a neutral sentiment. NLTK already has a built-in sentiment analyzer called VADER (Valence Aware Dictionary and sEntiment Reasoner) which calls the SIA function returns the compound polarity between the range of [-1,1]. VADER is best suited for the language used in social media (short sentences with slang & abbreviations). In 70% of the cases, both methods provide the same result. Also, both are more inclined toward the neutral sentiment. Mr. Musk would be happy to know that there is not much negative sentiment towards him. It is mainly neutral or positive.

Once we have the labeled dataset, I used the K-means clustering algorithm to check the sentiment cluster and the deep learning algorithm to classify the text. In K-Means clustering I used 3 clusters but before that, I created a TF-IDF matrix which refers to the Term Frequency–Inverse Document Frequency matrix. This matrix reflects the importance of a word to a document. In the deep learning model, I used a combination of neural networks such as CNN + RNN (LSTM). Convolutional neural networks excel at learning the spatial structure in input data. These learned spatial features may then be learned as sequences by an LSTM layer. We can easily add one-dimensional CNN and max-pooling layers after the Embedding layer which then feeds the consolidated features to the LSTM. The classification model presented the sentiments of each tweet. Each tweet was grouped into a class/category. The focus is on the deep learning model, which is designed to compete with the existing NLTK, Text Blob for the sentiment. The goal was to make my model at par with the existing open-source packages.

Evaluation and Final Results

The models were evaluated using the accuracy metric. The K-Means clustering algorithm had an accuracy of 64% whereas the deep learning algorithm using a neural network had an accuracy of 83%. The artificial neural network is comprised of node layers (consisting of neurons), containing an input layer, one or more hidden layers, and an output layer. Each node connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. If not, no data is passed along to the next layer of the network. Simple models are usually unidirectional so is the model in this project. For the future, we have an option to train the models in a bi-direction manner i.e., going from past to future and future to past. The information is passed back and forth, and the model can keep learning from its mistakes. This can improve performance over time. It is difficult to understand why the deep learning model including the convolution neural network (CNN) and recurrent neural network (RNN) with long short-term memory (LSTM) performs better because it is a black-box model. However, it is known what the model contains so that helps. The picture of deep learning architecture for this project is provided below,


embedding_1_input | InputLayer | input: [(None, None)] | output: [(None, None)]
embedding_1 | Embedding | input: (None, None) | output: (None, None, 100)
conv1d_1 | Conv1D | input: (None, None, 100) | output: (None, None, 1000)
max_pooling1d_1 | MaxPooling1D | input: (None, None, 1000) | output: (None, None, 1000)
lstm_1 | LSTM | input: (None, None, 1000) | output: (None, 100)
dense_1 | Dense | input: (None, 100) | output: (None, 1)

I believe that the preprocessing could be more robust because people do not necessarily use language properly on social media platforms. The language includes slang, short forms/ abbreviations, and emojis which the currently developed model could potentially consider with further development. Also, the sentiment classification could account for more emotions such as anger, love, happiness, sadness, and so on. Twitter is a significant platform but only one in five adults (22%) use it according to a 2019 Pew Research Center survey. Therefore, basing analysis solely on twitter sentiment is not the right way to go. Potentially, scrappers and API can fetch data from other platforms and combine the sentiment tally to get a broader picture. In the field of Text Analytics, the quality of text and size of the text is equally important. 50K data points of tweets may not necessarily be better than 10k data points of tweets if the data is of poor quality. However, the more the data more the possibility to build a bigger vocabulary and potentially better model. For this project, the amount of data seemed sufficient.

In the labeling part, the polarity scores obtained from both labeling methods were converted to sentiment groups. For this project range of polarity was set to balance the accuracy and sample size. Polarity score <= -0.4 was set as negative, polarity score >= 0.4 was set as positive, and range of (0.4,0.4) was set as neutral. Although I fetched 50k tweets, only 35.5k tweets had the same labels from NLTK VADER and textblob so I lost some amount (14.5k tweets) for analysis. I could have chosen one method over the other. If I had to, I would choose VADER/SIA due to the ability for better analysis of short text and the proportion of sample size (neutral: 32k, positive 12k, negative: 6k) whereas for textblob method (NLTK), (neutral: 41.5k, positive: 16k, negative: 7.3k). Neutral seems to be dominant in both labeling methods. When I did choose only one method, the accuracy was below 50%. The best accuracy achieved was when I took the tweets which were classified the same from the SIA method from NLTK and textblob. It proves the impact of accurate labeling for the model. Also, the dataset needs downsampling/upsampling because there is an imbalance of data for different sentiments. Tackling this issue could further improve the model further (combined model contains neutral: 29.6k, positive: 4.5k, negative: 1.3k).

Overall, the project can be concluded to be successful as it provides a decent demonstration of the application of deep learning models in the field of NLP. The paper has justified the reasons for the paths chosen for implementation.

Source and Literature

For achieving the goal, the python packages used were: NLTK, Textblob, sklearn, Keras, TensorFlow, NumPy, pandas, tweepy, and requests. The inspiration to develop the project was taken from the following papers and articles:

1. Y. Chen and Z. Zhang,” Research on text sentiment analysis based on CNNs and SVM,” 2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA), 2018, pp. 2731-2734, DOI: 10.1109/ICIEA.2018.8398173.

2. D. Goularas and S. Kamis,” Evaluation of Deep Learning Techniques in Sentiment Analysis from Twitter Data,” 2019 International Conference on Deep Learning and Machine Learning in Emerging Applications (Deep-ML), 2019, pp. 12-17, DOI: 10.1109/Deep-ML.2019.00011.

3. Mrunal Sawant, Text Sentiments Classification with CNN and LSTM, https://medium.com/@mrunal68/text-sentiments-classification-with-cnn-and-lstm-f92652bc29fd

4. Abdul Hafeez Fahad, Sentiment Analysis — Let TextBlob do all the Work!, https://medium.com/red-buffer/sentiment-analysis-let-textblob-do-all-the-work-9927d803d137