Understanding Polarity of Social Justice Tweets using Deep Learning Techniques

Mitchell Li
Hersh Solanki

Abstract

Given recent social events, understanding the change in sentiment polarity surrounding a specific event is beneficial in understanding how the world is perceiving them. Social media is a powerful platform that millions of people use to voice their opinions. Twitter allows users to categorize their voice towards certain events or movements through their hashtag(#) syntax. The general goal of the project is to use various natural language processing and deep learning techniques to analyze tweets surrounding controversial social events to understand how the general public reacts to those events as well as providing various methods in understanding the exact reaction through word clouds and graphs.

1. Introduction

It seems like every day new social events take the world by storm. The reception of some events is fairly partial toward one side, resulting in a general consensus that something is going to change for better or worse. However, most of the time, the reception is not so blatant. Understanding general sentiment on social events, reporting those responses, and responding to the event based on that sentiment can prove invaluable for a world so divided on important issues.

Twenty years ago, this would have been a near impossible task. But with the rise of social media, the impossible is now possible. Every day, millions of people use social media to voice their concerns on social topics to where people can respond or like what was voiced. There is now a concrete repository of daily general sentiment through these social media platforms.

Primarily, this paper will focus on Twitter, because it has the capability of tagging posts with their hashtag('#') notation. This effectively groups posts, making pulling data around certain social issues simpler. For example, for recent issues surrounding "Black Lives Matter", a script can be written to pull tweets with only "#BlackLivesMatter", eliminating the need to write a separate filtering script.

Now that the availability of data for sentiment analysis and technology to handle the computing tasks is present, we have developed a pipeline consisting of three main parts. The first component is a trained model on 1.6 million labeled tweets to predict sentiment on actual tweets to return the general, daily reception of specific hashtags. In addition, a configurable searching algorithm to allow for the pulling of tweets around dates to be analyzed. Lastly, a word cloud is generated from the most polar positive and negative tweets to give the user an idea of what exactly is being said.

2. Related Work

2.1 Sentiment(specifically Twitter) Analysis

The idea of sentiment analysis in natural language processing is nothing new. Prior to

the rise in popularity of deep learning, neural networking, architectures, the task was handled fairly well through Naive Bayes and SVM. [4]

As machines developed higher processing power, deep learning grew in popularity. As opposed to traditional machine learning techniques, deep learning uses decision trees to come to a conclusion. This design is actually inspired by human networks in the brain, hence the term, neural networks. By learning the correct weights to put on these neural connections, accurate predictions can be drawn from these complex trees of connections.

Convolutional Neural Network(CNN) and Recurrent Neural Network(RNN) architectures began to outperform previous iterations of machine learning modules. These new CNN architectures began with predicting Yelp and Google Reviews sentiment via multi-layered deep learning architectures containing one or more word embeddings.[3] However, performing sentiment analysis on reviews is a much more straightforward task than understanding the sentiment of tweets in social media, since the communication of sentiment is not as direct in social media. Which is where our project is able to take sentiment analysis to a more complex level in trying to understand sentiment in social media posts.

2.2 Emoji Sentiment Analysis using Neural Networks

Throughout the past decade, emojis have become a social norm in textual communication. Emojis themselves have evolved from the simple ":)" to the unicode representation of an actual "smiley face". As such, the need to understand emojis in natural language processing has grown more than before. [1]

As neural networks have become the norm of sentiment analysis, convolutional neural networks and recurrent neural networks have been used to understand emojis. Emoji embeddings have been created as well, and preliminary analysis on the accuracy of those embeddings has been conducted. However, this is where we offer a novel approach. Using a concatenated word and emoji embedding on real twitter data to understand sentiment has not been widely published; much less, a pipeline to analyze the results of the sentiment analysis.

2.3 History of Social Justice Hashtags

#BlackLivesMatter is the 3rd most used social-issue hashtag in Twitter's history. Hashtags are generally used to highlight important ideas within tweets that demand attention. From 2013 to 2016, #BlackLivesMatter was used almost 12 million times. As a response to this hashtag, #AllLivesMatters rose up. In the same time period, it was used 1.5 million times. To many critics, #AllLivesMatter was used to minimize the minority struggle. About 10% of the tweets for #BlackLivesMatter were critical of the movement, while 33% of #AllLivesMatter were critical of that movement. Ultimately, #BlackLivesMatter has generally been used significantly more positively then #AllLivesMatter. [2]

3. Data

As mentioned earlier, the training data that we used is a data set known as "Sentiment140". It was a labeled dataset created by Stanford students. The data set is completely cleaned of typos and emojis, making it ideal for training and validating

word embeddings. The set consisted simply of:

Sentiment	Hand labeled sentiment (positive or negative)
Id	The unique ID of the twitter post
date	The date of creation of the tweet
Device	The device that posted the tweet.
User	The twitter handle of the person who posted the tweet
Tweet	The text of the tweet

Table 1: Metadata of the Sentiment140 dataset.

Of these 6 categories provided in the dataset, the two of interest are Tweet and Sentiment. Using the data from these two columns, we are able to train our model to learn the correct labeling of sentiment of these 1.6 million tweets.

Since Sentiment140 did not contain emojis or typos, we could not confidently conclude that our model was robust enough to handle them. So, we pulled 1000 tweets from twitter and hand labeled them as positive or negative and ran a validation set against that newly labeled set to determine if the model was robust enough.

For the Tweets, we first used Tweepy, a Python wrapper to the Twitter API. However, this approach was not viable, because there is a 30 day limit of pulling tweets from the Twitter Search API. Instead, we used a package called GetOldTweets3. Using a JSON provider, this package can access tweets much deeper into the past. GetOldTweets3 allowed us to query on various hashtags, and set specific date ranges for the tweets. We then pulled daily tweets and understood the polarity of them using multiple deep learning models. In the analysis section, we discuss the importance of understanding these daily trends over time.

4. Methods

4.0 Training

A P100 GPU on Google Colab Pro was used to train the models. Normal CPU computation is far too slow to extensively train any deep learning models. Access to a state of the art GPU allows for efficient training.

4.1 CNN

A popular method in sentiment analysis is the use of word embeddings in multi-layered neural networks. However, an issue is that an emoji integrated neural network has not been widely published as of yet. The main one is known as “emoji2vec”. The creators of emoji2vec “maps emoji symbols into the same space as the 300-dimensional Google News word2vec embeddings. Thus, the resulting emoji2vec embeddings can be used in addition to 300-dimensional word2vec embeddings in any application.” [5] Although the completed embedding was not created, the individual embeddings of the 300-dimensional Google News word2vec and the newly trained emoji2vec can be concatenated into one embedding and used.

The emoji2vec uses the same mapping as the 300-dimensional Google News word2vec which was created with gensim. Although there are many word embedding vectors, we were specifically limited to gensim for that reason. The concatenated emoji and word embedding was created with an extra bin which we

labeled as the unknown bin. All of the words not recognized by the word embedding will be placed into this bin to allow for the sentiment analysis to continue while not completely losing those words. Once the word embedding was concatenated, a reverse mapping dictionary was created to allow the for reverse look up.

The CNN itself was fairly simple. We built a generic CNN with Keras. There was a

3 epochs with a batch size of 128. Surprisingly, the model performed pretty well. The training accuracy was 0.8394, and the validation accuracy was 0.78.

4.2 RNN

With the exact same word embedding used in the CNN, an RNN model was designed and trained to see if the results would improve in a recurrent network. A similar architecture was used where the layers are a text embedding layer, LSTM layer, and a dense layer. The default learning rate for Keras is 0.001. However, a learning rate of 0.0003 was used since the primary implementations of the model seemed to be learning too quickly, causing large, increasing validation losses in early epochs.

The RNN was trained over 3 epochs with a batch size of 128, similar to the CNN. The RNN performed about as well as the

e CNN. We hypothesized that by adding that RNN layer, relationships could be lost as we added more complexity to the layers.

4.4 BERT

We then began to explore BERT, Google's Bidirectional Encoder, which is the state of the art of NLP techniques. BERT attempts to understand the context of sentences via

single 1-dimensional convolution layer using a "relu" activation function. The output of the convolution was fed into a Max Pooling Layer. The final layer is a Dense layer.

The model was trained on a large proportion of Sentiment140 (about 1.58 million tweets). A validation set was created with the remaining 2000. The model was trained over

CNN. The training accuracy was 0.8068, and the validation accuracy was 0.7805.
This was not too surprising since the CNN performed that well to begin with.

4.3 CNN with RNN Layer

Since both the CNN and RNN performed about the same, we thought a good way to enhance each model's performance was to actually combine the two models. An LSTM layer was added into the CNN architecture. Using the exact same concatenated word embedding, the training was performed on this "stacked" model.

The results were actually worse than both of the individual models. The training accuracy was 0.813, and the validation accuracy was 0.759. This was a bit surprising, because we thought that adding another layer could only help the performance of the bas
masking, where 15% of the words are hidden and then predicted with the 85% of the words. BERT is rare in that it is bidirectional, meaning context is derived from words before and after the masked words. We are using a prebuilt architecture from Bert Base Uncased[1]. The model is pre-trained on raw text, with goals of masked

[1]We utilized a portion of pre-existing code from Chris McCormick and Nick Ryan published here: https://colab.research.google.com/github/orico/NLP/blob/master/BERT_Fine_Tuning_Sentence_Classification.ipynb

language modeling and next sentence prediction. BERT has 12 layers and over 110 million parameters

We first trained BERT on base parameters with 100,000 training samples of Sentiment140. The BERT training time is far greater than that of CNN/RNN given complexity of layers and complexity. It took 2 hours to train the model. We experimented with freezing layers in the BERT model to assist in the processing time for training the model, but the accuracy penalty it caused led us to not go further with freezing any layers.

Next, with a smaller data set, we spent time hypertuning various parameters of BERT. Our final results were trained with these parameters in mind. When testing our results, we achieved 88% validation accuracy, which was the highest of any of the models.

Moreover, it was important to have a class balance when training. With 80% of the dataset having a negative label, this was affecting our inference on the positive tweets. Thus, we made sure to sample equivalently from both classes before training the model.

5. Analysis

We chose BERT over the other models for a variety of reasons. First, the BERT model was the deepest with the most trainable parameters, leading to the best results. We can see an accuracy table below. Moreover, BERT has been state of the art in NLP models for the last year, using reasonable compute to derive high veracity results. Because of the focus on context of words, BERT is preferred to basic deep learning techniques.

Deep Learning Method	Validation Accuracy	Labeled Tweet Accuracy
CNN	0.78	0.65
RNN	0.78	0.68
CNN with RNN Layer	0.76	0.68
BERT	0.88	0.66

Table 2: Performance table of each neural network architecture.

We choose to understand both the hashtags “All Lives Matter” and “Black Lives Matters, as they provide information about opposing viewpoints. Taking a time series view, the average polarity of tweets are taken on July 17, 2014, March 28, 2020, and July 15, 2020. These correspond to the deaths of Eric Garner, Breonna Taylor, and George Floyd. A 15 day offset is used in order to understand polarity before and after the event. The results are as follows:

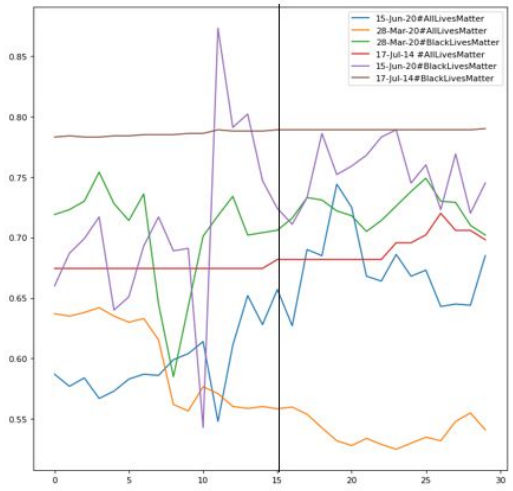

Figure 1: Line graph showing the daily polarity of tweets.

The line at the 15 day offset represents the day of the event. Across the board, other than #AllLivesMatter on March

28, 2020, there is a general downward trend following the event that results in an upward trend after. This signifies the tweets began negatively closely following the actual event, and result in general positivity after the event.

One hypothesis was that certain hashtags would come back to pre-event polarity rates, but this is not the case. There is a secular trend either up or down. We could have further evaluated these tweets on a longer time scale in order to understand these trends.

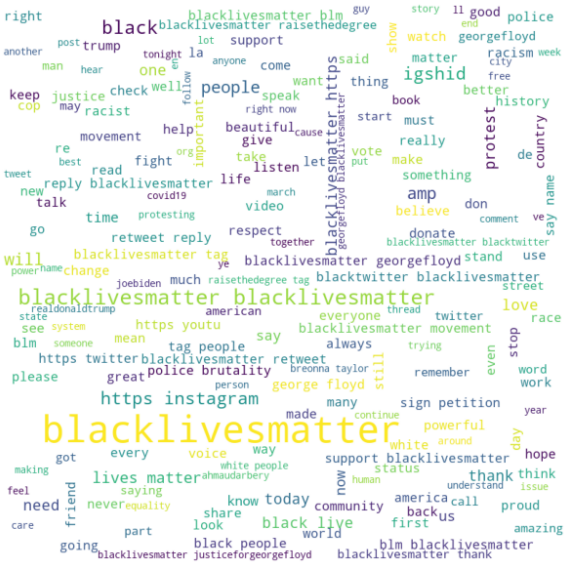

Figure 2: Word cloud generated from the most positive #BlackLivesMatters tweets surrounding June, 15, 2020.

In the pipeline, we finished the analysis with 2 word clouds for whatever hashtag was analyzed for the most positive and negative tweets separately. In Figure 2, a word cloud was created from a list of the most positive tweets from #BlackLivesMatters on the 30 days surrounding June 15, 2020. The size of the word represents the amount of times it showed up in the tweets. Some of the words that show up the most include donate, fight, black, proud, and amazing. In general, the words have positive and reinforcing connotations, which adds to the assurance that the model is performing and categorizing the tweets well.

6. Conclusion

In conclusion, a rigorous trial and error phase of different neural networking architectures were tried and tested, including the furthering of an existing emoji concatenated word embedding. Although there is quite a bit of exploration being done in sentiment analysis, we believe our novel approach in tying the custom word embedding to various neural network architectures and comparing the results to BERT prior to nominating BERT as the chosen model and creating our pipeline around it is completely unique.

The resulting pipeline, given the training and validation accuracies along with the results from the word cloud, creates accurate, aggregate analyses of tweets or any social media post. This is powerful for decision makers of any realm to use as a resource to assist in their decisions.

Therefore, we have shown that social media sentiment analysis can be performed to a high level of accuracy on multiple validation points. We also generated a pipeline around that NLP model to analyze historical social events to portray the general reception of those events through a positivity rating and a word cloud.

7. Data Release and Access to the Logic

https://github.com/mitchellcli/Tweets-Sentiment-Analysis-using-Deep-Learning-Techniques

Please cite us when using this logic for reference.

8. References

1. Anbukkarasi, S., and S. Varadhaganapathy. “Analyzing Sentiment in Tamil Tweets Using Deep Neural Network.” *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, 2020, doi:10.1109/iccmc48092.2020.iccmc-00084.
2. Anderson, Monica. “History of the Hashtag #BlackLivesMatter: Social Activism on Twitter.” *Pew Research Center: Internet, Science & Tech*, Pew Research Center, 31 Dec. 2019, www.pewresearch.org/internet/2016/08/15/the-hashtag-blacklivesmatter-emerges-social-activism-on-twitter/.
3. Baad, Dipika. “Sentiment Classification Using CNN in PyTorch.” *Medium*, Towards Data Science, 6 Apr. 2020, towardsdatascience.com/sentiment-classification-using-cnn-in-pytorch-fba3c6840430.
4. Chen, Yuxiao, et al. “Twitter Sentiment Analysis via Bi-Sense Emoji Embedding and Attention-Based LSTM.” *2018 ACM Multimedia Conference on Multimedia Conference - MM '18*, 2018, doi:10.1145/3240508.3240533.
5. Eisner, Ben, et al. “emoji2vec: Learning Emoji Representations from Their Description.” *Proceedings of The Fourth International Workshop on Natural Language Processing for Social Media*, 2016, doi:10.18653/v1/w16-6208.