Emojify This: The GSB CS230 Experience

Shammi Quddus, Srishti Sundaram, Ruchir Shah
{shammi,srishti, ruchirfs}@stanford.edu
Winter 2018

Abstract

We use a dataset of tweets with emojis from an online twitter archive to train a model that predicts the likely emojis for a given tweet. Tweets with emojis are used to train a bidirectional LSTM with dropout, using the open source GloVe Twitter word vectors which have been trained on a Twitter corpus. Our results show a "Top 1 emoji" accuracy of about 10.7% on the test set. We discuss methods to improve upon this result. Sample outputs from our model include:

CS230 is my favorite class. ❤️ ✨ ❤️ 🙂 ♀

I love Stanford. ▶️ ❤️ ✨ ⚽ ❤️

Can we meet for coffee?. ✨ ❤️ ♀ ⚽ ❤️

Fig 1: Sample output from our model

1. Introduction

Sentiment analysis is an important, yet-to-be-perfected problems in natural language processing. Sentiment analysis can have myriad applications for companies that have large volumes of textual consumer data for example in the form written reviews, comments, chat conversations. In addition, companies that analyze large volumes of voice like Alexa, Siri and Google, sentiment analysis can provide companies information to improve customer experience and product recommendations.

Many existing efforts that have achieved success in sentiment detection only detect binary sentiments: happy or sad. However, for a machine to truly communicate like a human, it is critical to understand a wide range of different emotions. Applications for such a tool are widespread, ranging from automated rating in online reviews to building speech agents that can understand human emotions.

Emojis are an effective means through which humans indicate sentiment in everyday communication; using emojis as a labeled proxy for sentiment could be instrumental in unlocking the next level of success in sentiment detection. Tweets with emojis are unique as they usually summarize or add to the emotion associated with a short length of text. A model that predicts the appropriate emojis for a given piece of text is analyzing sentiment, using emoji as a proxy for sentiment. Our

2. Related Work

This project was inspired by the work done by Rahwan et. al at the MIT Media Lab in the Deepmoji project, where they analyzed 1.2B billion tweets to train a deep learning model to detect emotions and sentiment using emojis. Their 128 unit LSTM model had a top one accuracy of 17%, which, in context, is quite significant considering the number of potential emojis that could have been selected. The seminal DeepMoji project has achieved considerable success in transferring the results of the study to a variety of test data sets far outside the Twitter sphere, including literature and works of art. In addition to conducting an in-depth study, the Deepmoji project has even provided a simple-to-use consumer facing tool that allows anyone to enter a sentence for the model to predict the most likely emojis for that sentence.

3. Dataset and Features

We used a large, publicly available set of JSON Twitter data from November 2017. Downloading and pre-processing raw data took significant effort.

- **Removing Metadata:** The Twitter data contained a substantial amount of metadata apart from the actual text of the tweets. In our experimentation, we decided to focus only on the words of the tweet instead of using any of the other provided data (tweet data, location, userID, etc.). We made this decision, because we want our end model to be testable on a wide variety of resources beyond solely Twitter. In the real world, if we want to assess sentiment in everyday sentences and phrases, we need to use the words of a sentence and their order - only - and ignore all other Twitter metadata.

- **URL and Foreign Language:** In the second stage of pre-processing, we further cleaned up tweets to remove data that was irrelevant to our study. First, we removed URLs and usernames from the data; usernames, in particular, were quite prevalent since many tweets are in fact retweets of other users's posts. As a part of this stage, we also removed any non alphanumeric characters including punctuation and characters of non-Western scripts. Our focus is initially on just Western text.

- **Non-emoji tweets:** Lastly, we removed all tweets that did not contain emojis.

- **Separating text and emoji:** We separated out the text (X) of individual tweets from the emojis(Y) used in them. If a tweet contained multiple emojis, we create separate data points for each use. We made this decision to credit tweets that used multiple emojis, since the authors of these tweets likely felt stronger about their emoji usage than those authors who

only used one emoji. After separating text and emojis, we indexed the emojis to assign numbers to each one; our data set contains 149 unique emojis.

- **Tokenization:** Lastly, we tokenized the X and passed each word through a 50 dimensional pre-trained Twitter GloVe model to generate embeddings.

- **Unbalanced dataset:** We analyzed a total of about 4M tweets, but our final, cleaned, data set includes approximately 7,322 tweets Heart emoji was found in 30% of our datasets followed by sparkles emoji which was about 25%. Our model learned that the best way to maximize accuracy is to guess heart emoji and it will be statistically correct 30% of the time, regardless of meaning. Initially we were very excited to hit such a high accuracy but when we printed our results we saw that the first emoji was always heart by default. We had to manually hand engineer our dataset to make it more evenly distributed with no single emoji exceeding 300 observations in our dataset.

The small size of our dataset is a very significant limitation of our project and we underestimated how much time it would take to clean up data. Radwan etl. al used millions of tweet to train for each single emoji. Due to small data size we only have a train and test set split by 90%-10%. We decided on this larger proportion of training data, because our data set is quite small and we wanted to ensure adequate data points for training.

4. Methods

We used a sequential baseline and a bidirectional LSTM structure implemented in Keras. The LSTM model was built using source code from Stanford University's CS230 course.

Baseline: Our baseline model has two fully connected layers, one hidden layer and one output layer. The hidden layer uses a ReLu activation function with 64 hidden nodes. The output layer uses a sigmoid activation and contains 15 nodes.

Final: Our final model is shown in the figure below. Each bidirectional LSTM layer has 128 nodes with tanh activation, each dropout layer is set to 50%, with a fully-connected softmax activation layer. We print the top five probabilities from the softmax layer. We use a cross-entropy loss function, an adam optimizer and accuray as our performance metric when compiling the model.

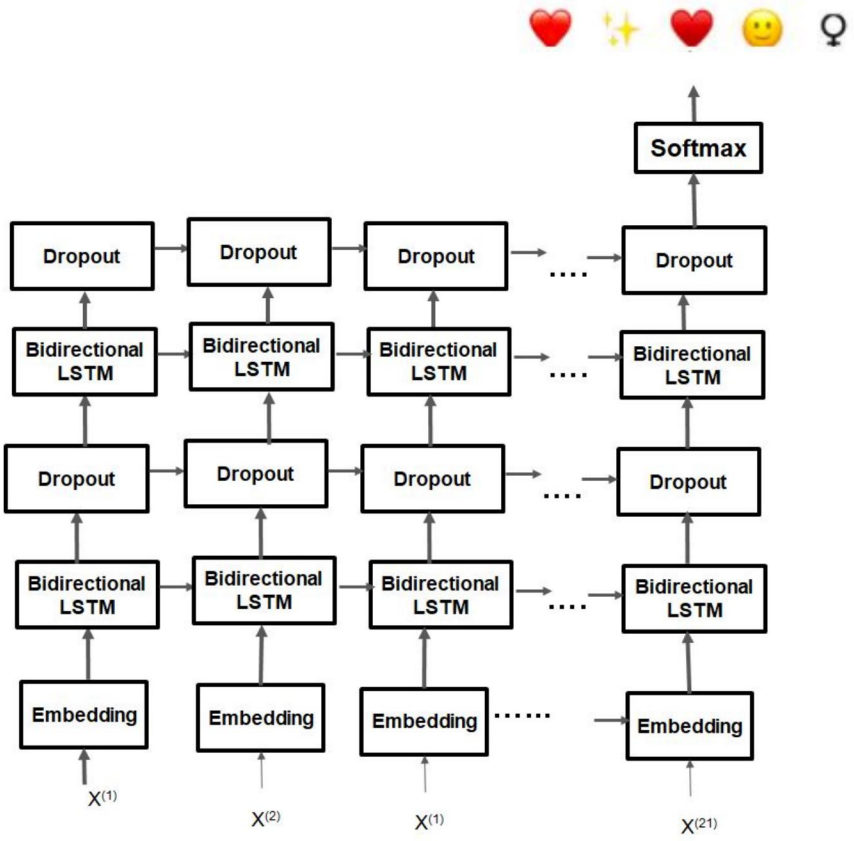
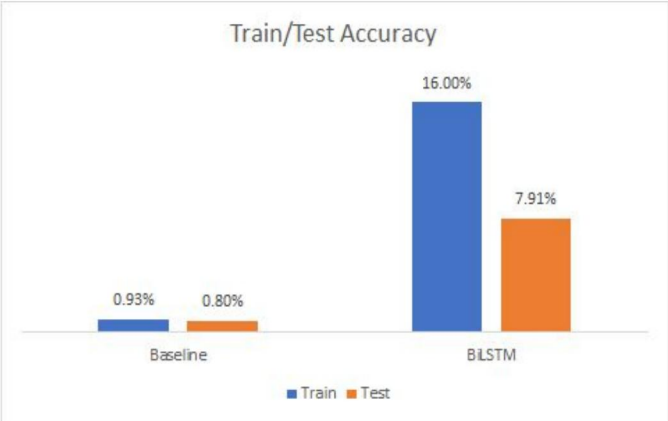


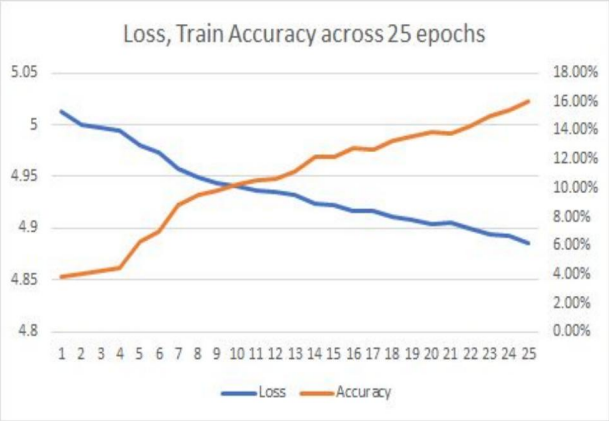
Fig: LSTM Model

5. Experiments/Results/Discussion

After running the model on our training data, we printed softmax results of the top five emojis for every single tweet of our test data. However, to calculate accuracies, we used "Top-1" accuracy, only counting a data point as accurately predicted if the top returned softmax percentage was equal to the correct emoji. Here is an example of some sample outputs below.

Our results showed that the sequential, bi-LSTM model performed substantially better over the baseline, non-LSTM model. Accuracy improved substantially on both the test set and on the training set. Using our bi-LSTM model, we were able to achieve test results of 7.91%, which is quite impressive considering our data set consisted of 149 unique emojis. Our train accuracy is 16% and the difference of about 8 p.p. shows that we have a certain degree of overfitting.





6. Conclusion/Future Work

While we were able to validate some key findings in this study, there remains significant room for growth. As a team, we were limited by the time taken to create a significantly large dataset. We underestimated how much of the dataset we would be able to keep. As we pursue this project in the future, we plan to examine the following.

First, we would like to test our existing model on a much larger data set. We believe a much larger data set (MM of tweets) could dramatically increase accuracy. This would entail access to much more substantial computing power.

Next, we would like to continue fine-tuning our existing model to improve results; in particular, we would like to explore integration of one or more attention layer(s) to improve our model's parameters.

Lastly, perhaps the most interesting area of exploration would be a generative model. We have been able to successfully generate the correct emoji when given a tweet. Can we do the opposite - could we create original tweets based on emoijs? Future work could examine whether GANS or a similarly generative model might be utilized to synthesize sentiment-focused text from an input emoji.

7. Contributions

Every teammate managed a primary module and helped others as when needed. Srishti acquired the Twitter data and cleaned it up along with significant leadership on the model. Shammi's primary responsibility was to build the two models and she helped Srishti with cleaning the data. Ruchir set up the initial Word2Vec, worked with AWS, and managed the final deliverables. .

8. References

ArchiveTeam JSON Download of Twitter Stream 2017-11.
https://archive.org/details/archiveteam-twitter-stream-2017-11

Bjarke Felbo, "Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm," Media Lab, Massachusetts Institute of Technology. Oct. 7, 2017.

Francesco Barbieri, "Are Emojis Predictable?". Large Scale Text Understanding Systems Lab, TALN Group. Universitat Pompeu Fabra, Barcelona, Spain. April 3-7, 2017

9. IPython notebook

Google drive link to notebook:
https://drive.google.com/file/d/10Xcs3xM98RbTUaxrk-pe2098DWPWWxLj/view?usp=sharing