CS230

Public Forgetfulness? Reputation Visualization in the Wake of #MeToo

Michael Cai
B.S. in Computer Science
Stanford University
mcai88@stanford.edu

Alex Goodman
B.S. in Symbolic Systems
Stanford University
alexgood@stanford.edu

Abstract

[Content Warning: Sexual Assault] In this project, we used a Twitter sentiment classifier to visualize the conversations surrounding celebrities accused of sexual assault in the wake of the #MeToo movement. The task of sentiment classification of tweets is notoriously difficult due to both the brevity of the form and the common use of nonstandard spellings or slang terms. To perform the classification, we fine tuned a deep convolutional neural network trained on the Sentiment140 database in order to classify tweets as positive and negative, achieving an F1 score of **0.853** on our test set. We then used the classifier to visualize how the sentiment of tweets about accused celebrities changed over time as a result of the accusations.

1 Introduction

During the ongoing #MeToo movement that reached a fever pitch last summer, many celebrities had their reputations drastically altered (rightfully so) by accusations of sexual assault. In this project, we built a neural sentiment analyzer to visualize how the conversations about accused celebrities on Twitter have changed as a result of those accusations by classifying tweets about those celebrities as positive or negative.

For this project, we trained a deep convolutional neural network (CNN) using the Sentiment140 tweet dataset along with pretrained GloVe Twitter 27B word embeddings to make a classification decision. This was an interesting challenge because tweets are generally less structured compared to other forms of text, and will often contain slang or intentional misspellings that do not appear in any corpus. In addition, the brevity of tweets (maximum tweet length is 140 characters) can make it difficult to extract sentiment from so little information.

The architecture for the model we ultimately selected was designed by Michael in his CS224N project, in which he compared a number of model architectures for classifying Twitter data. For this CS230 project, we improved upon the original model by performing a more extensive hyperparameter search, and then applied the model to our main focus issue of reputation visualization for celebrities accused of sexual assault.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

2 Related work

2.1 Neural Sentiment Analysis Models

The task of sentiment analysis is one of the oldest and most common tasks in natural language processing. Numerous methods have been tested for solving this problem, such as Naive Bayes and SVMs. However, as with most other machine learning tasks, the past few years has seen a shift away from traditional machine learning methods in favor of more powerful neural network architectures.

The use of convolutional neural networks for sentiment analysis was explored in depth by Yoon Kim, who found that CNN models could perform just as well, if not better, compared to more traditional RNN and LSTM models (1). State of the art performance on several sentiment analysis tasks was achieved through Deep CNN architectures first introduced by Facebook Research's Conneau, et. al. in 2017 (2). Through the use of a multi-layered CNN and character level word embeddings, the Facebook team was able to classify the polarity of longer form text - Yelp and Amazon reviews - with 95.72% accuracy. However, their model was not tested on data from Twitter. Our model was heavily inspired by this model, with the primary difference being the use of GloVe word embeddings rather than character level embeddings. We felt that this model was particularly clever because it is powerful enough to perform very well on sentiment tasks, but wasn't so heavy that it required too much computational power to train. Google AI Research's BERT model would eventually set the state of the art benchmarks for most NLP tasks such as sentiment classification using a very deep LSTM network trained across multiple Google TPUs (3). However, our project lacked the time and computational power to fully explore the application of BERT to this task.

2.2 Classifying Twitter Sentiment

In our research, we found a few attempts at classifying Twitter sentiment using various methods. In their exploration of SVMs for sentiment analysis, Mohammad, et. al. collected tweets labeled positive, negative, and neutral from several sources including the Sentiment140 database referenced in this project (4). The group extracted features from each tweet such as character n-grams, number of hashtags, emoticons, etc. for their classification and achieved a F1 score of 69.02 in their 3-way classification. Similar work was performed by Barnes, et. al. in their assessment of text classification techniques. The group trained multiple models on the SemEval dataset for the 3-way classification of Tweets, and achieved the best results using a Bi-LSTM model, with an F1 score of 68.5 (5). Unfortunately, we was unable to find any literature on the exact task that we are working on for this project, which makes the project all the more interesting.

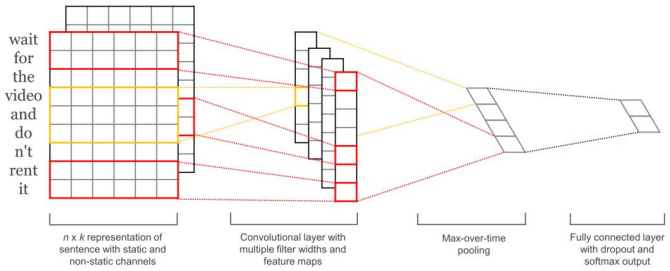
3 Dataset and Features

We trained our model on the Sentiment140 dataset collected by Stanford graduate students (link: `http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip`) (6). It is free to use and contains about 1.6 million tweets labeled as "positive" or "negative", which we split into a training set of about 1.5 million tweets, and a dev set of about 100,000 tweets. The zip download also includes a test set of 500 tweets, which we used for our final model evaluation. While documentation online says the dataset contains "neutral" tweets as well, upon observation, we found very few (only about 150) tweets that were labeled as "neutral," so we did not consider them in our model.

For the purposes of our experiments, we adjusted the dataset so that the "negative" tweets were given the label "0" while the positive tweets were given the label "1". The few neutral tweets we did find in the dataset were all given a label of "1" (so in a way, our model is being trained specifically to identify negative tweets). For example, the tweet *@kenburbary You'll love your Kindle2. I've had mine for a few months and never looked back. The new big one is huge! No need for remorse! :)* was given a label of 1.0, while *@Karoli I firmly believe that Obama/Pelosi have ZERO desire to be civil. It's a charade and a slogan, but they want to destroy conservatism* got a label of 0.0. These decisions were made to simplify the tasks of calculating training loss and model accuracy, allowing us to use a simple single-output sigmoid layer to perform the final classification rather than a multi-class softmax layer. Finally, all tweets were processed using the pretrained GloVe Twitter 27B 100d vectors which can be found here: `http://nlp.stanford.edu/data/glove.twitter.27B.zip` (7).

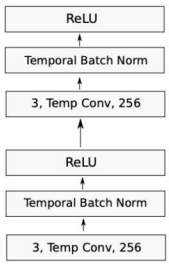
4 Methods

The technique of convolving data in neural network originally arose from the task of image processing, but was later adapted for text classification tasks due to its ability to extract contextual features surrounding each word. A 1D convolution involves multiplying word vectors for a window of text by a matrix of weights called a kernel. The results of the convolution can then be processed through max pooling, which selects the maximum weight in each window of a vector.



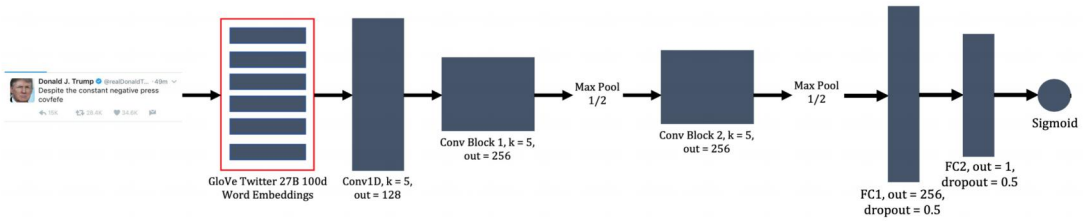
A 1D convolution with max pooling and an FC layer. Source: `http://www.wildml.com/`

The deep convolutional neural network architecture we constructed was heavily inspired by the work of Conneau, et. al. (2) The crux of our model is the convolutional block design outlined by Conneau, et. al. which contains two 1D Convolution - Temporal Batch Norm - ReLU sequences, a depiction of which is reproduced below.



The convolutional block architecture. Source: Conneau, et. al.

This we coded using Pytorch (8) to function as its own discrete class, allowing for maximum reusibility in expanding our neural network depths. For both 1D convolution layers, we used a kernel size of 5, padding of 2, and stride length of 1, and for the Max Pool layer, we used a kernel size of 2. The result of each convolution block is that the number of channels is doubled, and the length of each input sentence is halved. Our model utilized two of these conv-block/max pool layers followed by two max pool layers and a sigmoid function that outputs a value between 0.0 and 1.0 for the final classification. Finally, we used a binary cross entropy loss function, which is given by the equation $-(y \log(p) + (1 - y) \log(1 - p))$.



A 2 Conv Block model with GloVe Twitter 27B 100d word embeddings

In order to retrieve the data to which we wanted to apply our model, we opened Premium Developer accounts with Twitter (9) to access the full tweets archive. We wrote a Python script that pulled 600

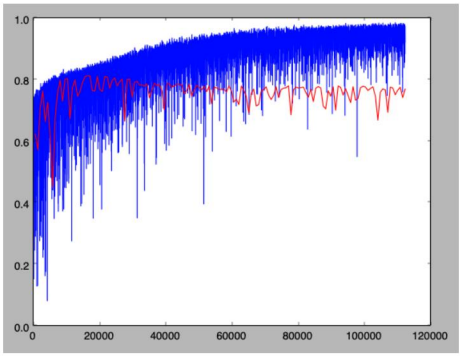
tweets about each alleged perpetrator from right after news of the allegations broke and from the present. After cleaning those tweets and removing emojis in order to match the training distribution, we fed them into our trained model to see how public sentiment has changed in the time since news first broke about each person.

5 Experiments/Results/Discussion

We evaluated our model using two metrics commonly utilized in literature to evaluate sentiment classification: the binary accuracy score and the F1 score, given by the following equations:

$$accuracy = \frac{Correctly\ Labeled\ Examples}{Total\ Examples}, F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

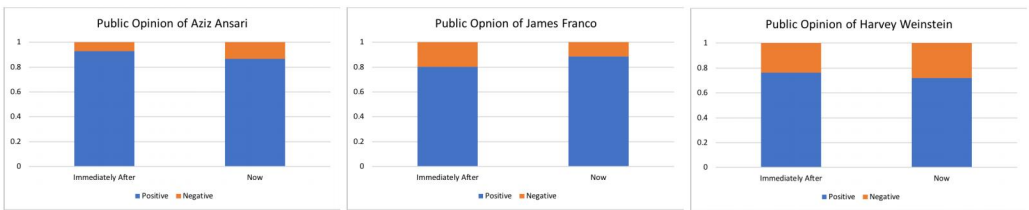
We noticed that our model was very strongly overfitting our training data. For example, when we compared the F1 scores achieved on the training set vs the dev set over time for our base model, we obtained this graph:



Thus the main hyperparameter we decided to focus on was the weight decay. In other words, the loss function for a given input x, true output y, predicted output $\hat{y}$, and Cross Entropy Loss function L, is now given by the equation $L2 = L - \lambda||(w)||^2$. We tested several values of λ, the results of which can be found below. However, we found that in all cases, weight decay hurt model performance, so we ultimately decided to use early stopping to optimize our model to fight against overfitting.

Model	**Top Dev Set F1 Score**
Weight Decay = 1.0	0.730
WD = 0.1	0.668
WD = 0.0001	0.807
WD = 0.00001	0.805
Early Stopping, no WD	**0.817**

We found that results were mixed about how public sentiment changed around these public figures. For some, public opinion was lower immediately after than it currently is, suggesting that the public did at least partially forget. For others, though, public opinion is currently worse than it was immediately after news broke of the allegations against them. We include here such examples examples of our results.



6 Conclusion/Future Work

In the future, we would like to explore fine tuning and maybe adding our own layers on top of the BERT neural architecture to achieve true state of the art performance on Twitter sentiment classification. Additionally, we would like to obtain better versions of our dataset about the celebrity perpetrators because limitations of the Twitter API hurt our ability to get as much data as we wanted in the way that we needed.

7 Contributions

Michael architected each of the models we tried based on his work for CS224N, while Alex worked on hyperparameter search for those models and data collection and visualization through the Twitter APIs.

8 Code

Our github repository can be found here:
`https://github.com/theCaiGuy/Twitter-Sentiment`

References

[1] Y. Kim, "Convolutional neural networks for sentence classification," Sep 2014.

[2] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for text classification," *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Jan 2017.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," oct 2018.

[4] S. M. Mohammad, S. Kiritchenko, and X. Zhu, "Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets," 2013.

[5] J. Barnes, R. Klinger, and S. S. I. Walde, "Assessing state-of-the-art sentiment models on state-of-the-art sentiment datasets," *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 2017.

[6] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," 2009.

[7] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[8] "Pytorch."

[9] "Premium search - twitter developers."