Twitter Sentiment Analysis using Deep Learning Models

Jiaying Guo, Ang Li, Yuan Li, Yujie Sun, Shiping Yi

Abstract

Hundreds of Millions of tweets are sent each day, and it has become necessary to study twitter sentiment analysis as people are locked at home due to Covid19 and they would most likely to express their feelings using twitter. This project aims to develop models for twitter sentiment analysis. We develop different models using deep learning, and use the models to study Covid19 tweets.

1 Problem Definition

Sentiment analysis is the extraction and classification of the emotions of a sentence or text that has wide applications in various areas. There have been many breakthroughs with Natural language processing techniques that have paved the way for commercial advertising, social media monitoring of elections, as well as detecting criminal activity from user tweets.

In this project, we aim to explore using CNN models to train a set of processed labeled twitter sentiment dataset, and compare them with the baseline in literature. To extend the scope, the baseline model will be explored in conjunction with LSTM network and chaning word embeddings. The best model will be applied to the task of monitoring the overall sentiment of twitter users during the time of the Covid-19.

2 Literature Reviews

This paper (Allem et al., 2017) evaluates the importance of debiasing twitter data when extracting sentiments. The author uses data pulled from the Twitter API using the root terms related to "Hookah" and discriminates between the tweets of automated social bots and real human writers by comparing three main features –timing of tweets, spam labels, ratio of tweets from mobile versus laptop1. Some other features used are information diffusion pattern, friends features, content, and sentiment features2. These features were then fed into a model combining Support Vector Machine algorithm and rule-based reasoning and automated inference logic to classify sentiments. Grammatical structures such as the modifier "very" and "but" are used to adjust sentiment scores. The model is trained on 4 SemEval and the ISEAR emotion datasets (Mohammad et al., 2016) and an emotion-tagged tweet (Gou et al., 2014) corpus.

Pagolu, V. S.et. al.(Pagolu et al., 2016) manually built twitter text dataset for sentiment analysis by applying techniques like tokenization, stop-words removal and regex matching for special character removal for raw data pre-processing, and classified tweets into three categories: positive, negative and neutral. For feature extraction and text representation, they selected N-gram and 300d word2vec representations. They experimented with three types of machine learning models, random forest, logistic regression and SMO, for analyzing the correlation between stock market movements of a company and sentiments in tweets. On the whole, they found out that models trained with word2vec representation gives more sustainable and promising performance over large datasets. Among the models, the random forest models with word2vec representations achieved the highest accuracy of 70.2%.

S Rosenthal et. al.(Rosenthal et al., 2019) automatically filtered the tweets for duplicates and removed those for which the bag-of-words cosine similarity exceeded 0.6, retaining only the topics for which at least 100 tweets remained. Then they used CrowdFlower to annotate the new training and testing tweets. In terms of methods, the use of deep learning stands out in particular, including neural network methods such as CNN and LSTM

networks. Supervised SVM and Liblinear were also very popular, e.g. combining SVM with neural network methods or SVM with dense word embedding features. Classifiers such as Maximum Entropy, Logistic Regression, Random Forest, Naive Bayes classifier, and Conditional Random Fields were also used by other teams. Their primary measure for sentiment analysis is AvgRec (average recall), which is averaged across the POSITIVE (P), NEGATIVE (N), and NEUTRAL (U) classes.

Severyn, A et. al.(Severyn and Moschitti, 2015) first gained their word embeddings for tweets through training a neural language model on unsupervised tweets dataset, follow a 3-step process to initialize the weights of their network: 1) use word2vec model to learn the word embeddings on an unsupervised tweet corpus 2) use a distant supervision approach with their convolutional neural network to refine the embeddings 3) use the parameters obtained from previous step to initialize the network. Their neural network architecture for sentiment classification is a single convolutional layer followed by a non-linearity, max pooling and a soft-max classification layer. They use stochastic gradient descent (SGD) to train the network and backpropagation algorithm to compute the gradients. For tuning the learning rate, they apply the Adadelta update rule. To avoid overfitting, they augment the cost function with l2-norm regularization terms and apply dropout when computing the activations at the softmax output layer. Their approach sets a new state-of-the-art on the phrase-level and ranks 2nd on the message-level subtask of Semeval-2015 Twitter Sentiment Analysis (Task 10) challenge.

The author in this paper (Kim, 2014) used word embeddings trained from unsupervised learning models–word2vec, to train a series of Convolutional neural Networks with some additional modification such as having separate channels for static word vectors and task-specific word vectors (fine-tuned via backpropagation). The model consists of one layer of convolutional layer convolved with multiple filters and a max-over-time pooling operation for different feature extraction, followed by a fully-connected softmax layer to output probability distribution over the labels. The embeddings used are vectors trained by (Mikolov et al., 2013) on 100 billion Google News words. L2-norm regularization method was used to prevent overfitting. The model is tested on several datasets including movie reviews, Stanford Sentiment Treebank 1 and 2, as well as TREC (question dataset), and CR (customer reviews). The results show that models fed with the pre trained word embeddings outperform the state-of-the-art models on 4 out of 7 tasks including sentiment analysis and question classification.

Rojas-Barahona, Lina Maria (Rojas-Barahona, 2016) applied non-recursive neural networks, recursive neural network and combination of both methods for sentiment analysis, movies reviews corpus (Pang Lee, 2005) were used as training and testing dataset. Among the 11 methods, DCNN (Kalchbrenner et al., 2014) reached the highest accuracy of 86.8% for binary classification. CTree-LSTM(Tai et al., 2015) reached the highest fine-grained of 51.0 with an accuracy of 88.0. Sentiment analysis in twitter was also done. Tweets were annotated with the polarities positive, negative, and neutral. The combination of sentiment-specific word embeddings and NRC-ngram reached the highest fscore of 86.48%.

The paper (Stojanovski et al., 2016) proposed a model consisting of two neural networks: a convolutional neural network(CNN) with a single filter and a gated recurrent neural network(GRNN). Words are mapped to word embeddings using GloVe embeddings pretrained on the Common Crawl dataset with a dimensionality of 300. The convolutional operation is applied to every possible window with window size of 3, followed by max-over-time pooling operation. GRNN makes use of sequential data and modulates the flow of information inside the gating unit. Both networks output a fixed size vector. They are concatenated to a single feature vector, which is fed into a softwax layer. The softmax regression classifier gives probability distribution over labels in the output space. The label with highest possibility is the final prediction. Dropout regularization with a dropout parameter of 0.25 is used because too many parameters are being learned. Results are evaluated using subtasks from (Nakov et al., 2019). Different evaluation metrics are used for different tasks. Subtask B has an F1 of 0.748, subtask D has

the best KLD with 0.034 and task E has an EMD of 0.316.

The authors (Ramadhani and Goo, 2017) are Adyan Marendra Ramadhani and Hong Soon Goo from Dong-A University. The title of the paper is Twitter Sentiment Analysis using Deep Learning Methods and it is published in 2017 7th International Annual Engineering Seminar (InAES). The paper used the dataset from the stream API with 1000 good dataset (both Korean and English), 1000 bad dataset (both Korean and English) as training dataset (without words Themes) and 2000 test dataset (both Korean And English). The Deep Neural Network used has the following parameters: Feedforward Neural Network, 3 Hidden Layer, Using the Rectified Linear Unit (ReLU) and the sigmoid function activation, The input is 100 neurons, Using Mean Square Unit and the Stochastic gradient descent. The first hidden layer is filtering the words, the second layer is filtering based on the sentence and the third layer is based on the popularity of the words based on the online dictionary. Based on the graph the training model accuracy is going up constantly and the test gets the convergence on the 55%. The table shows the accuracy of the train and test set. With the result of 77.45 % and the 75.03 %.

Since we have constructed our data from merging t4sa_text_sentiment.tsv and raw_tweets_text.csv to a new csv file based on tweet id, we have include the code in the code section.

3 Experimental Design

Your Experimental Design section should include a description of your training data, your evaluation metric, and your simple baseline model along with its performance on the test set. You can adapt your Milestone 2 submission for this part. (300-500 words, plus 2 figures/tables, plus 1 or more equations).

We select multi-class twitter sentiment data as our project's dataset, and we clean up and split up our data into three different datasets for training, developement and testing. F1 score, recall and precision will be our major evaluation metrics, and we tried out majority class baseline model for our data to see whether the data is suitable for our further experiments. In the future, we hope to develop more sophisticated models and apply our best model to manually labeled covid-19 twitter sentiment test dataset.

3.1 Data

Twitter for Sentiment Analysis(T4SA) dataset has been used for training and testing dataset generation. These tweets were selected from a larger dataset of 3.4 million tweets by most confident textual sentiment predictions. The data given in http://www.t4sa.it/ contains id and text of all the collected 3.4 M tweets and the textual sentiment classification of the 1,179,957 selected tweets of the T4SA dataset. We apply regular expressions methods to replace special format like "@2pmthailfans", urls and dates with our predefined string to avoid confusion, since the original dataset only give the probability results of the three classes, we select the most likely one among the three classes (NEG: negative, NEU: neutral, POS: positive) to be the data's final label: -1 for negative, 0 for neutral and 1 for positive class.

3.1.1 Examples of original data

id,text,TWID,NEG,NEU,POS

768097627686604801,Josh Jenkins is looking forward to TAB Breeders Crown Super Sunday https://t.co/antImqAo4Y https://t.co/ejnA78Sks0,
768097627686604801,
0.00808970047903,
0.04233099488469999,
0.9495793046359999

768097631864102912,RT@2pmthailfans: [Pic] Nichkhun from krjeong86's IG https://t.co/5gcAcu9by7,
768097631864102912,
0.014643660775799998,
0.9265568133679999,
0.0587995258562

3.1.2 Examples of processed data

label, text

0,RT AT_USER I think this how boys that ask for nudes see me as URL

1,just woke up

3.1.3 Dataset size

TS4A was split into train,dev and test datasets.

Dataset Size	Train	Dev	Test
T4SA	884,967	58,998	23,992

Table 1: Table of train, dev and test dataset size

3.2 Evaluation Metrics

fscore would be selected as the main evaluation metrics to score system outputs. Precision and Recall would also be taken into consideration as secondary evaluation metrics. fscore would be calculated based on the predicted labels (-1 = negative, 0 = neutral, 1 = positive) and the actual labels.

3.3 Simple Baseline

3.3.1 Random Baseline

The first simple baseline is ramdom baseline. For each text, randomly lable as -1=negative, 0=neutral, 1=positive.

Dataset	Train	Dev	Test
Precision	0.3341	0.3297	0.3316
Recall	0.3342	0.3289	0.3308
F1 Score	0.3150	0.3102	0.3125

Table 2: Table of published baseline model results on T4SA dataset

3.3.2 Word-Count Baseline

The second baseline is a stronger one. Spilt each text into single words, count the number of positive words and negative words. If the number of positive words is larger, label as 1=positive, if the number of negative words is larger, label as -1=negative, if equal, label as 0=neutral.

Dataset	Train	Dev	Test
Precision	0.7340	0.7354	0.7353
Recall	0.6347	0.6338	0.6359
F1 Score	0.6641	0.6635	0.6654

Table 3: Table of published baseline model results on T4SA dataset

4 Experimental Results

4.1 Published Baseline

We re-implement the state of the art model as proposed by Severyn, A et. al.(Severyn and Moschitti, 2015). for the Semeval-2015 Twitter sentiment analysis challenge. It's a multi-class single-layer convolutional neural network with 300 filters of size 10 with dropout and L2-regularization, using 300d word2vec representations for our T4SA dataset to train and evaluate with adadelta adaptive learning rate optimizer. Due to the problems of getting approved for our twitter developer accounts, we couldn't use the same dataset as described in the paper in time, so we switch to our T4SA dataset. We use F1 score as our evaluation metric, after training for 5 epochs, our results on training, validation and test set are as follows:

Dataset	Train	Dev	Test
Precision	0.8855	0.9396	0.9372
Recall	0.8774	0.9304	0.9060
F1 Score	0.8812	0.9104	0.9200

Table 4: Table of published baseline model results on T4SA dataset

It turns out our results performs slightly better than that of the published baseline (F1 Score = 0.8479). Although our test set is different from that of the paper, our dataset size is much larger (as shown in the following table compared to the ones in the paper), which accounts for more cases in twitter sentiment. However, due to the same problem, our results are not directly comparable to that of the paper.

Dataset Size	Train	Dev	Test
T4SA	884,967	58,998	23,992
Twitter'13 (paper)	5,895	648	2,734

Table 5: Table of comparing dataset size of our model and that of the published model

4.2 Extension1

In this section, we use other word-embedding features to train the network and compare performance. We choose to do this because we want to know besides changing hyperparameters, whether it possible to improve the fscore with precision and recall higher than the published baseline. There are several pretrained word embedding such as GloVe(Global Vectors for Word Representation), FastText and Google word2vec, with different structures. We found that with FILTER SIZES $= [3, 4, 5]$ and word embedding GoogleNews-vectors-negative300, we can achieve an fscore of 0.96 which is higher than the published baseline. We also include a table of results, where the entries are the performance of the baseline or one

of your extensions and the captions are the model structures and word-embedding features to train the network.

In this extension, we did experiments where we searched over a set of different parameters and model structures and use other word-embedding features to train the network. Here, we include a result on how varying the parameter changed the performance on the test set.

filters size	[5]	[5,5]	[3,4,5]	[5,4,3]
word2vec	0.9200	0.9357	0.9454	0.9697
6B100D	0.8812	0.8923	0.9051	0.9123
6B300D	0.9093	0.9169	0.9222	0.9433
42B300D	0.9457	0.9571	0.9697	0.9658

Table 6: Table to test different model structures and word-embeddings

4.3 Extension2

For extension2, we feed the otuput of the Convlutional layer to the input of one LSTM layer. The idea is that while the convolutional operation can be useful to extract the features of the sentence, using a LSTM network can keep the temporal dimension of the original sentence. While LSTM network is used commonly for sequence prediction and image captioning, it is used here as a means to preserve the temporal relationship within the sentences. The results are shown in table4.

It was expected that adding more layers of LSTM with dropout would potentially increase the overall training and testing fscores. However, in this case adding more LSTM layer does not indicate better performance.

Layers	Train fscore	Val fscore	test fscore
1	0.8813	0.9104	0.9200
2	0.8816	0.8678	0.8880
3	0.8558	0.8573	0.8844

Table 7: Table for performance of adding lstm layers after Convolutional layers

When altering the model architecture by first feeding the embedded vocabulary into LSTM layers first, and then use the output as input of proceeding Convolutional layer followed by max pooling and a fully connected layer(same as the baseline model) has proven to increase performance of the original baseline. The fscores are shown below in table5.

Consistent with the CNN+LSTM architecture, adding more layers does not guarantee improvement in performance. No obvious overfitting was seen with the first two layers added, but there was significant plateau in validation loss even with the added dropout rate of 0.1. Introducing one layer of LSTM before CNN network may yield the best results in terms of fscore among all other configurations experimented. This model performs slightly better than the baseline.

Layers	Train fscore	Val fscore	test fscore
1	0.9244	0.9173	0.9388
2	0.8434	0.8719	0.9119
3	0.2356	0.2315	0.2320

Table 8: Table for performance of adding lstm before convolutional layer

4.4 Result Sum Up

Summary the best results of models.

Model	fscore
Random Baseline	0.3125
Word-Count Baseline	0.6654
CNN(Published Baseline)	0.9200
CNN+word-embedding(Extension1)	0.9697
LSTM+CNN(Extension2)	0.9388

Table 9: Table for performance on test dataset using different models

From the result above, random baseline had the lowest fscore, word-count baseline had a higher one. CNN(published baseline) showed a great improvement on fscore, the fscore of LSTM+CNN(Extension2) was a little higher that of published baseline. CNN+word-embedding(Extension1) had the highest fscore of 0.9697.

4.5 Extension3

For extension3, we manually annotated tweets of Covid19 of past few months. Each tweet was annotated by three people independently. The annotation which most annotators agreed were chosen as the label of the tweet. 772 tweets were annotated and tested using the models described above.

From the result above, random baseline had the lowest F-1 Score, word-count baseline had a higher one. CNN(published baseline) showed a great improvement on F-1 Score, the F-1

Model	F-1 Score
Random Baseline	0.3489
Word-Count Baseline	0.4507
CNN(Published Baseline)	0.5983
CNN+word-embedding(Extension1)	0.6007
LSTM+CNN(Extension2)	0.5930

Table 10: Table for performance on Covid19 dataset using different models

Score of LSTM+CNN(Extension2) was similar to that of published baseline. CNN+word-embedding(Extension1) had the highest F-1 Score of 0.6243. The F-1 Score order was almost the same as that of using T4SA test dataset.

4.6 Error analysis

4.6.1 T4SA Dataset

In this subsection, we perform an error anlaysis for our best performing system. Show examples of the errors that it makes. We show examples of the errors that the published baseline makes that our extensions get correct (and vice versa).

1. Fundraiser at The Greene Turtle Football FAMILY Fun URL (predicted label:0, true label:-1)

2. China July industrial profits rise, buoyed by increased sales, falling costs URL URL(predicted label:1, true label:0)

3. URL : 100 Inbound Marketing Content Ideas Part 1-5 URL(predicted label:-1, true label:0)

4. New Listings: Homes for Sale in and around Woodridge URL URL (predicted label:-1, true label:1)

5. URL $0.99 gt; ES-71II Camera Replacement Lens Hood for Canon EOS EF 50mm f/1.4 USM Lens $0. URL (predicted label:-1, true label:0)

6. RT AT_USER [PIC] 160821 MONSTA_X Kihyun IM - MTV Asia Music Stage in Taiwan [CR: CrushOn2226] URL (predicted label:-1, true label:1)

From this, the model seems to prone to predict -1, that is, overly pessimistic when it went wrong.

4.6.2 Covid19 Dataset

It is obvious that the fscore shows a huge drop when the test dataset changed from T4SA dataset to covid19 dataset. Here are two possible reasons. First, the models were not trained using Covid19 related dataset, a same expression might have different emotional tendency under different topics. Second, the annotations might be not accurate enough. For instance, it would be hard to define a tweet reporting new Covid19-death cases as neutral or negative. It would be possible that two out of three annotators annotated such a tweet as neutral while it actually should be negative based on the state-of-art annotation rules.

1. AT_USER electionnight BGC16 myvote2016 LUCKY FAN ur gonna win Hillary, praying for you URL (predicted label:0,true label:1)

2. love you too URL (predicted label:1,true label:0)

3. Called it. I said if they don't start winning they will have to trade people..... URL (predicted label:0,true label:1)

References

Jon-Patrick Allem, Jagannathan Ramanujam, Kristina Lerman, Kar-Hai Chu, Tess Boley Cruz, and Jennifer B Unger. 2017. Identifying sentiment of hookah-related posts on twitter. *JMIR public health and surveillance*, 3(4):e74.

Liang Gou, Michelle X Zhou, and Huahai Yang. 2014. Knowme and shareme: understanding automatically discovered personality traits from social media and user sharing preferences. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 955–964.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. Semeval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani, and Veselin Stoyanov. 2019. Semeval-2016 task 4: Sentiment analysis in twitter. *arXiv preprint arXiv:1912.01973*.

Venkata Sasank Pagolu, Kamal Nayan Reddy, Ganapati Panda, and Babita Majhi. 2016. Sentiment analysis of twitter data for predicting stock market movements. In *2016 international conference on signal processing, communication, power and embedded system (SCOPES)*, pages 1345–1350. IEEE.

Adyan Marendra Ramadhani and Hong Soon Goo. 2017. Twitter sentiment analysis using deep learning methods. In *2017 7th International Annual Engineering Seminar (InAES)*, pages 1–4. IEEE.

Lina Maria Rojas-Barahona. 2016. Deep learning for sentiment analysis. *Language and Linguistics Compass*, 10(12):701–719.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2019. Semeval-2017 task 4: Sentiment analysis in twitter. *arXiv preprint arXiv:1912.00741*.

Aliaksei Severyn and Alessandro Moschitti. 2015. Twitter sentiment analysis with deep convolutional neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 959–962.

Dario Stojanovski, Gjorgji Strezoski, Gjorgji Madjarov, and Ivica Dimitrovski. 2016. Finki at semeval-2016 task 4: Deep learning architecture for twitter sentiment analysis. In *Proceedings of the 10th International workshop on semantic evaluation (SemEval-2016)*, pages 149–154.