A Comparison between DataStories and BERT Classifier for Sentiment Analysis

Xiao Zhang
Leiden University
z764104102@outlook.com

Yanfang Hou
Leiden University
y.hou.2@umail.leidenuniv.nl

ABSTRACT

In this paper we focus on the subtask A of SemEval-2017 Task 4: identity the sentiments (positive, negative and neutral) expressed in the given tweets. We investigate the two deep learning methods for this task: one is called DataStories, which is a deep LSTM with attention presented by Baziotis et al. [2], the other is the BERT classifier from [9]. Then we do a comparison analysis among the two methods and the two baselines (Naive Bayes and Logistic Regression). The experiments show that the two deep learning models produce better classification results than the baselines and the BERT classifier performs slightly better than DataStories.

KEYWORDS

sentiment, LSTM, attention, BERT

1 INTRODUCTION

Sentiment analysis is a natural language processing technique which can be used to recognize the sentiment expressed in text. It has been widely performed for different types of texts, for instance, retailers learn about what the customers like or dislike about the products through sentiment analysis on the product reviews; by analyzing the sentiment hidden in the news articles, people could obtain the news opinions for certain topics. With the growth of social media, Twitter becomes increasingly popular with young people, politicians and general public. It is a 'microblogging' system which allows people to express their opinions by sending short posts called tweets or retweeting messages posted by others. So as a huge platform, the tweets in Twitter containing a large amount of emotions, opinions and attitudes are potentially helpful in sentiment analysis research. In this paper, we focus on the sentiment analysis on Twitter data, specifically the subtask A of SemEval-2017 Task 4 [8]: for a given tweet, our task is to identify whether the message is of positive, negative or neutral sentiment.

Many supervised machine learning and deep learning approaches are used to address this problem by treating it as a classification problem. According to the summary of the participants' results presented in [8] , two deep learning methods DataStories [2] and BB_twtr [4] achieve the best performance with the macro F1-score reaching 0.681. In the traditional machine learning methods, INGEOTEC [7], SiTAKA [6] and UCSC-NLP [1] perform well with the macro F1-score higher than 0.6. Besides, the latest significant development in Natural Language Processing (NLP) is the release of BERT [5], which has widely used in different NLP tasks including sentiment analysis.

We are interested in the two deep learning methods, one is DataStories, which uses Long Short-Term Memory (LSTM) networks with two kinds of attention mechanisms [2], the other is the BERT classifier that fine tunes the pretrained BERT model for classification [9]. Both of the two methods uses no hand-crafted features or sentiment lexicons but at the expense of deep networks and complex models. However, the LSTM based method trains the model from the scratch on the data, whereas the BERT based method fine tunes the pretrained model on the data. Therefore, although the BERT model is larger, it might be faster to reach a good result than LSTM.

To better understand the Twitter sentiment analysis problem and the performance difference between the LSTM and BERT model for this problem, our task mainly include the three parts: (1) replicate the DataStories method; (2) apply the BERT classifier used in [9] to our Twitter data; (3) do a comparison between the two methods in terms of performance.

The rest of this paper is organized as follows. In Section 2, we discuss the works that relate to sentiment analysis. Section 3 describes the Twitter dataset that we use and do a exploratory data analsis. Section 4 briefly explains the two methods and describes the baseline and evaluation used in the experiment. In section 5, the experimental results on DataStories, BERT classifier and baselines are provided to evaluate the performance. Finally, we discuss the work and conclude the project.

2 RELATED WORK

Many studies conduct sentiment analysis in a supervised way. Traditional machine learning methods are commonly used to address this problem. Jabreel et al. builds a Support Vector Machine (SVM) classifier with a novel set of features, which includes basic word features, syntactic features, the information provided by the lexicons, embeddings and clusters [6]. Abreu et al. also trains a SVM classifier but uses an ensemble of multiple models with different combination of features [1]. Both of the two methods achieves good performance with f1-score reaching more than 0.6 in the subtask A of SemEval-2017 Task 4.

The performance of traditional machine learning in sentiment analysis are restricted by the quality of features, whereas the deep learning methods break the limitation by utilizing much deeper networks to automatically learn the features instead of manually extracting features. The model called DataStorie reaches the first rank among all methods in the subtask A, which trains a LSTM network with the attention mechanism for sentiment analysis. Compared with traditional machine learning, such a sequential model could better capture the contextual information in a sentence. Another method called BB_twtr also achieves excellent performance, which uses an ensemble of CNN and LSTM networks for this task. In 2018, Google develops a new language representation model called BERT, which has proven to be very powerful in a wide range of NLP tasks [5]. In terms of sentiment analysis, the BERT model is fine-tuned on the SST-2 benchmark, which is a binary sentiment classification task on movie reviews, and the results on the official

GLUE leaderboard [1] show that it greatly improves the previous deep learning networks such as LSTM and ELMo [10].

3 DATA

3.1 Tweet data

Each tweet can be up to 140 characters long but it not only contains text messages but also has symbols, mentions, hashtags and other metadata. Twitter has many features which can be helpful in sentiment analysis. Table 1 shows four important features of tweets. Many tweets contains website links which are initiated as a URL. Mention is used when the users want to refer to someone via "@someone". Hashtag usually refers a topic by "#topic". Emoticon can be really helpful in sentiment analysis. It imitates the facial expressions of human which can greatly reflect the users' emotion.

Table 1: Main features in Twitter

Features	Explanation
URL	website link
Mention	@, refer to someone
Hashtag	#, refer to text/topics
Emoticons	:), facial expressions

Table 2 shows three tweets and its corresponding sentiment in the dataset. It gives a direct look about the tweet data. The three tweets contain some features such as Hashtags, Mentions, URL and Emoticons. And they also show some properties of tweets: short, unorganized and informal, which make the sentiment analysis challenging.

Table 2: Three tweets and corresponding sentiment

Tweet	Sentiment
Iran: Mass execution of 8 prisoners in just 3 days https://t.co/ljPpQ1l1Tv #health #politics #sun	Negative
I write a letter FOR you At THE meet on 30 APRIL 2016 Milan :) @JackJackJohnson	Positive
You may not talk to Justin bieber unless spoken to now hold on just a minute https://t.co/vpXNyHQtYG	Neutral

3.2 Data statistics

Train. The dataset are collected from previous year (2013-2016) containing development set, test set and train set. The table 3 shows the number of three kinds of sentiments in each dataset. we got 19903 positive tweets, 7840 negative tweets and 22591 neutral tweets. This shows there is an unbalance in dataset which we need to deal with in data processing. Note that we use all the above data for training.

Test. We use the same test set with the original paper [2]. It is a collection of 12284 English tweets, including 2375, 3972 and 5937 tweets with positive, negative and neutral sentiment respectively. We could see that the tweets that belong to the neutral class accounts for about half of all the tweets.

[1] https://gluebenchmark.com/leaderboard

Table 3: Statistics about training data from 2013-2016

Dataset	Positive	Negative	Neutral	Total
2013-dev	575	340	739	1654
2013-test	1475	559	1513	3547
2013-train	3640	1458	4586	9640
2014-sarcasm	33	40	13	86
2014-test	982	202	669	1853
2015-test	1038	365	987	2390
2015-train	170	66	253	489
2016-dev	843	391	765	1999
2016-devtest	994	325	681	2000
2016-test	7059	3231	10342	20632
2016-train	3094	863	2043	6000
Total	19903	7840	22591	50334

4 METHODS

4.1 Preprocessing

A tool called ekphrasis [3] is used for text preprocessing, including text tokenization, spell correction, normalization and segmentation. In ekphrasis, the well-designed tokenizer can recognize most emoticons, emojis and many kinds of expressions: date, time and more. For segmentation, it can split a long string to words. For spell correction, it can correct wrongly spelled words to the most possible word. In our text, we lowercase all the words and normalize the special expressions in the sentences such as urls, handles, date, emails and more).

4.2 DataStories

The method called DataStories obtains the 1st rank in subtask A of the SemEval-2017 Task 4. The model is based on LSTM network equipped with an attention mechanism [2].

Class Weights. In the dataset part, we found that the data is skewed, so we apply a weight to each class in the loss function. The weight is calculated by the formula 1 where x is the vector with the class counts. Table 4 shows the weights in the model.

$$W_i = \frac{max(x)}{x_i} \tag{1}$$

Table 4: Class Weights

Class	Weight
Neutral	1.0
Positive	1.13
Negative	2.87

Model. The model is constructed by BiLSTM equipped with an attention mechanism. LSTM is a special kind of RNN, which also works on the time series data and could keep track of the long-term dependency that the simple RNN could not address. It is capable of preserving information for a long period and combining it with the current input by adding gates and memory cell. The attention mechanism means that it will pay more attention to some specific

factors during training. Because not all words/sentences can express sentiment, using the attention mechanism helps to weight the importance of each word.

The whole model has three kinds of layers: Embedding Layer, BiLSTM Layers and Attention Layer. In the Embedding layer, the input words are projected into a low dimensional vector space. In our case, 300-dimensional pre-trained Glove word embedding is used to initialize the weights. In the BiLSTM Layers, the bi-direction performs well to get word annotations that contain information. In the Attention Layer, it assigns weight to each word annotation. The whole model is shown in Figure 1.

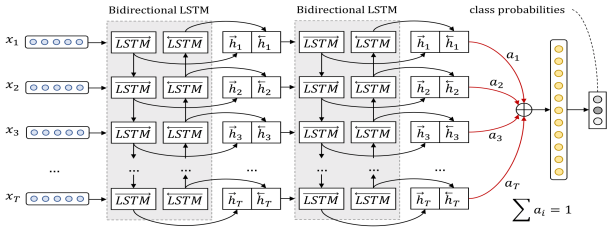

Figure 1: Bi-LSTM with attention,from [2]

Training. For the training process, we split the data into train set and validation set with 9:1 ratio. The most configuration is similar with original model but particularly we set less epochs for it can already generate good result in 5 epochs.

4.3 BERT Classifier

We follow the tutorial [9] to build a BERT classifier for sentiment analysis, fine tune the BERT model using our data and adjust the model parameters to fit our case. Specifically, the whole model mainly consists of the two parts: the first part is the BERT model, which encodes each sentence into the embedding vector, and the second part is a single-hidden layer feed forward network, which takes the sentence embedding generated by BERT as input and outputs the scores of the three sentiments for each sentence. We use the sentiment with the highest score as prediction and meanwhile the BERT embeddings are fine tuned during the training. The pretrained BERT model we use is called *bert-base-uncased*, which consists of 12 transformer layers and each layer produces the embedding vector for each token. The *CLS* is a special token added at the beginning of a sentence and we use its final hidden vector as the aggregate sequence representation, since it has been used by the BERT model for classification tasks and the experiments have proved that it is an effective sentence representation [5].

Table 5 shows the network structure and training parameters of the BERT classifier. Compared with the model used in [9], we add a dropout layer and use a smaller learning rate to prevent overfitting. Note that we split the entire training data into the train set and the validation set at a ratio of 9:1. The former is used for model training, and the latter is used for model evaluation so as to select the most suitable parameters.

4.4 Baseline

The two basic methods are used as the baselines.

Table 5: Model summary of the BERT classifier

Layers	Values
bert-base-uncased	Hidden units: 768
Linear	Output neurons: 50
ReLU	/
Dropout	Probability: 0.5
Linear	Output neurons: 3
Loss	Cross entropy
Optimizer	AdamW, lr: 1e-6
Epoch	5 times

- Naive Bayes: We follow the tutorial [9] to implement a Naive Bayes classifier. We first convert the text into a matrix of TF-IDF features for unigrams, bigrams and trigrams. Then these features are fed into the Naive Bayes classifier for training.
- Logistic Regression: We replicate the experiment of Logistic Regression by using the implementation provided by [2]. Each token is encoded into a 300-dimensional Glove embedding vector and the mean embedding vector of each sentence is used as features for logistic regression.

4.5 Evaluation

We evaluate the model performance by using the same metrics with the original papers [8] [2]. The tweets with the neutral sentiment account for a larger proportion on the test set. In this case, if the prediction tends to the majority class, the accuracy might still reach a high value but may not perform well on other classes. Thus apart from the accuracy, the measures based on precision and recall are also used for evaluation due to their robustness to data imbalance.

- Accuracy: the proportion of correct predictions.
- Average recall: the mean recall across the positive, negative and neutral classes.
- Average precision: the mean precision across the positive, negative and neutral classes.
- F_1^{PN}: the mean F1-score across the positive and negative classes.

5 RESULTS

Table 6 shows the experimental results of the four methods on the test set. Overall, both of the deep learning methods (DataStories and BERT classifier) outperform the two baseline methods (Naive Bayes and Logistic Regression). Among these, the BERT classifier achieves slightly higher scores than DataStories, while the Naive Bayes works worst.

We will discuss the four models in detail by combining the overall scores with the scores over the three categories shown in table 7. First, Naive Bayes has very low F_1^{PN} value, as a result of the extremely small recall score on the negative class. However the recall score of the neutral class is relatively high. These suggest that Naive Bayes predicts most tweets to have neutral sentiment, which is the majority class on the test set, thus boosting the other overall scores. Second, we notice that both of Logistic Regression and DataStories have a high recall score on the neutral class and a low recall score on the negative class, which indicates that the two models

incorrectly predict the many tweets as negative when they actually have the neutral sentiment. Finally, the performance of the BERT classifier is relatively balanced among the three classes.

Comparing with the original paper of DataStroies($F_1^{PN} = 0.675$), our implementation gets a slightly lower result because we just run 5 epochs rather than 50 epochs. However, this saves a lot of time as each epoch consumes about 20 minutes and the result is acceptable for further discussion.

Table 6: Results on the test set

	AvgRec	AvgPre	F_1^{PN}	Acc
Naive Bayes	0.468	0.695	0.270	0.529
Logistic Regression	0.636	0.620	0.634	0.607
DataStroies	0.669	0.658	0.669	0.648
BERT classifier	0.690	0.689	0.677	0.692

Table 7: Precision and recall over the three classes

	Positive		Negative		Neutral	
	Pre	Rec	Pre	Rec	Pre	Rec
Naive Bayes	0.564	0.517	1.000	2.518e-4	0.522	0.888
Logistic Regression	0.597	0.632	0.539	0.827	0.725	0.449
DataStories	0.657	0.647	0.578	0.847	0.741	0.515
BERT classifier	0.636	0.714	0.738	0.634	0.692	0.723

6 DISCUSSION

Result explanation. As we expect, the two deep learning methods performs better than the baseline, because they takes the sequential information as input whereas the baselines treat the text as a bag of words. Thus the DataStories and the BERT classifier could capture the contextual information of the word and the results also indicate that the BERT model does a better job than LSTMs in dealing with long-term dependencies. On the other hand, data imbalance is one important challenge in our task. Here, we discuss its impact on the four methods. First, the predictions of Naive Bayes are affected by the prior probability of the three classes, leading to a high recall on the majority class (neutral) and a low recall on the minority class (negative). To alleviate the influence of data imbalance, both of Logistic Regression and DataStories weight the loss of different classes. Concretely, the class weight is inversely proportional to class frequencies in the training data, which means in our case the incorrect predictions of the negative class will be penalized more than the neutral class. As a result, although the performance difference of the two methods over the three categories is not as large as the Naive Bayes, but the difference in scores between the neutral and the negative class is apparently greater than that of the BERT classifier. For BERT classifier, it does not do extra processing for unbalanced data, but it achieves a good balance among the three classes, indicating its strong robustness to the unbalanced data.

Method limitation. From the previous result analysis, we see that both Logistic Regression and DataStories tend to recognize the neutral text as negative due to heavier penalty on misclassification of negative text. Thus we could explore different class weighting methods to improve the result. Second, training a deep LSTM model with attention usually consumes much time, so we might try GRU model, which is also a variant of RNN but has fewer parameters than LSTM, to boost the training. For the BERT classifier, the pretrained BERT model we use is not trained on tweet data, hence it might be useful to try the pretrained models on the English tweets.

7 CONCLUSION

In this paper, we study the problem of sentiment analysis and compare the performance between DataStories, BERT classifier and the baselines. We found that the two deep learning models outperform the two baselines and the BERT classifier achieves slightly better performance than DataStories. For future work, we could apply the two deep learning model to new datasets in other fields and observe their performance.

8 CONTRIBUTION

Overall, we make equal contributions for this project. Xiao is mainly responsible for the reproduction of DataStories and Yanfang mainly focuses on the implementation of the BERT classifier.

REFERENCES

[1] José I Abreu, Iván Castro, Claudia Martínez, Sebastián Oliva, and Yoan Gutiérrez. 2017. UCSC-NLP at SemEval-2017 Task 4: sense n-grams for sentiment analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. 807–811.

[2] Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*. 747–754.

[3] Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. DataStories at SemEval-2017 Task 4: Deep LSTM with Attention for Message-level and Topic-based Sentiment Analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, 747–754.

[4] Mathieu Cliche. 2017. Bb_twtr at semeval-2017 task 4: Twitter sentiment analysis with cnns and lstms. *arXiv preprint arXiv:1704.06125* (2017).

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[6] Mohammed Jabreel and Antonio Moreno Ribas. 2017. SiTAKA at SemEval-2017 Task 4: Sentiment analysis in Twitter based on a rich set of features. In *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*. 694–699.

[7] Sabino Miranda-Jiménez, Mario Graff, Eric Sadit Tellez, and Daniela Moctezuma. 2017. INGEOTEC at SemEval 2017 Task 4: A B4MSA ensemble based on genetic programming for Twitter sentiment analysis. In *Proceedings of the 11th international workshop on semantic evaluation (SEMEVAL-2017)*. 771–776.

[8] Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 task 4: Sentiment analysis in Twitter. In *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*. 502–518.

[9] Chris Tran. [n.d.]. *Tutorial: Fine tuning BERT for Sentiment Analysis.* Retrieved January 10, 2021 from https://skimai.com/fine-tuning-bert-for-sentiment-analysis/

[10] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461* (2018).