Twitter Sentiment Classification

* A CSE842 Course Project

Chang Liu
Computer Science and Engineering
Michigan State University
East Lansing, MI
liucha39@msu.edu

***Abstract*—Text sentiment classification is a subset question of text classification which is a classical application of Naive Bayes and supervised learning. However, sentiment classification is not the same thing as typical text classification. This paper will utilizes a semi-supervised algorithm combining lexical analysis with machine learning methods.**

***Index Terms*—Classification, Text sentiment, Twitter text prediction**

I. Introduction

Text sentiment once could be impossible before the accent of social medias like Twitter or Facebook when relative dataset for analysis are quite few. Most newspapers hold neutral sentiment in their reports, although different perspectives. However, the sentiments in text are quite important because it's the very first step of artificial intelligence. There are many science fictions imagining the abuse of robotics and rebellion of AI. But these imaginations cannot be true only if AI understands human sentiments.

One bad news is that text sentiments are extremely complicated that even humans sometimes cannot understand others' words, let alone AI. For this reason, SentiWordNet [1] proposed a classical method which assigns a sentiment normalized value to individual words. For example, -0.625 for 'sad' and 0.875 for happy. The assignments simplifies the text sentiment as word sentiment which return a sentiment list for a sentence or paragraph. And then, we can generate labels for text by numeric methods such as Naive Bayes or Logistic Regression.

There are a obvious challenges here for traditional text sentiment classification. First of all, there might be several different sentiments in a paragraph or even in a single sentence. For example, 'Although nearly all members in our football teams gave up, John still fight and believe we can win this game'. The sentiment of the first half of the sentence is definitely negative, but the rest part is positive and the whole sentence can be neutral or positive. So single labels are quite unsatisfied for this kind of scenarios.

Many researches proposed different methods to improve the classical text sentiment classification. One of them dealt with the disambiguation in word sense [2]. It's common that many English words have multiple meanings with opposite sentiments. For example, for the word 'mouse', a positive sentence could be 'My mom give me a new mouse as the birthday present.' while the negative sentence could be 'There are mouses everywhere in my house, so annoying.' And this WSD algorithm [2] can also analyse contextual subjectivity besides word sense disambiguation.

Nowadays, machine learning is widely applied to text sentiment classification. There are several advantages here. Firstly, machine learning makes no conditional independence assumption between features while Naive Bayes usually fails for high conditional related text. Also, LSTM [3] and Reinforcement learning [4] are better at training contextual semantics of words which means even with very few training data, These algorithms can build good language models. However, machine learning techniques are quite time-consuming while easy to overfit. There are some other methods such as Attention-Based Modality-Gated Networks [5] which automatically focuses on the discriminative features for sentiment classification.

This paper will present a semi-supervised learning algorithm using lexical sentiment and attention Networks as extra features. Unsupervised learning may also be tried but I expect it to be a benchmark as Naive Bayes since the metrics may be unstable for different text.

II. Related Work

There are 3 main types of the sentiment text analysis: knowledge-based techniques, statistical methods, and hybrid approaches [6]. The Knowledge-based techniques utilize the properties of each word or phrase such as 'happy', 'sad', 'angry' which can be found in most twitter messages because people always tweet and comment certain kinds of things. The statistical methods don't focus on the prior knowledge that we have because one sentence can be consisted of many words with different or even opposite opinions and we need a statistical algorithm to parsing all these sentiment to get the overall sentiment or opinion of the whole text. Bt the way, the hybrid methods take advantages of the above 2 methods by combing, for example, the knowledge based labels and statistical models like neutral network.

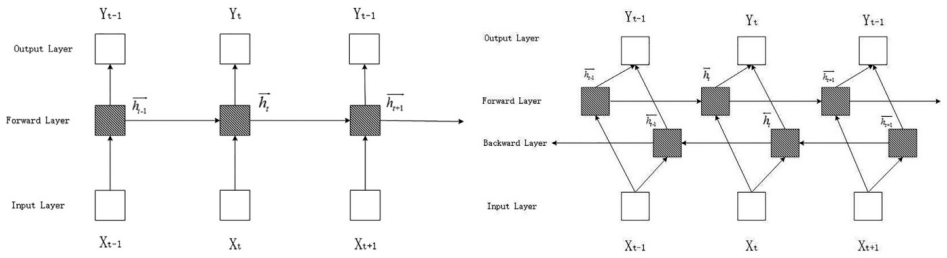


Fig. 2. Illustration of a LSTM model (a) and a BiLSTM model (b).

Fig. 1: LSTM structures
Figure source [7]

	john	likes	eating	apples	hates
sentence 1	1	1	1	1	0
sentence 2	1	0	0	1	1

TABLE I: Bag of Words examples

III. Methods

A. Bag of Words

Before analyse the models used in this paper, we should first talk about the bag of words which return a matrix disregarding grammar and even word order but keeping multiplicity. For example, there are 2 sentences *john like eating apples.*' and *john hates apples.* can be first transformed into a mapping dictionary as *john:0, like:1, eating:2, apples:4, hates:5* which index is just the vocabulary of all sentences. And then it can be written into a matrix such as table I shown.

B. Baseline Models

There are 2 probabilistic classifiers in this paper used as the base line: Naive Bayes and Logistic Regression. Since both these two models have rather simple structures and a few hyparameters (learning rates for Logistic Regression), the performance will be quite stable and robust. However, the simple structure can easily lead to underfitting which means the model can not fit the training models and hard to improve anyway.

C. Probabilistic Classifiers

In this paper, I will go through multiple prob classifiers such as: SVM, KNN and Decision Tree (Including Naive Bayes and Logistic Regression). All these models utilize the bag of words techniques which will be introduced in detail at experiment parts (including the TFIDF and PCA feature extraction).

1) SVM: support-vector machines techinque is meant to maximize the distance at hyperplane for different classes. While a hyperplane can be written as $\omega^t x - b = 0$, parameter $\frac{b}{||\omega||}$ determines the offset of the hyperplane from the origin along the normal vector ω

2) KNN: the k-nearest neighbors algorithm is a non-parametric method proposed by Thomas Cover which classify objects by its neighbors. If $k = 1$, the KNN algorithm degrades to the single nearest neighbour algorithm. Meanwhile, if k equals the numbers of total samples, our algorithm behaves as underfitting and it gives a smooth decision surface and everything becomes one class which is the majority class in our DataSet. Generally, as K increases, our decision surface gets smoother.

3) Decision Tree: DTs are a non-parametric supervised learning method used for classification and regression. A decision tree starts at a root node and branches in two or more directions recursively to form a `tree`. Each branch offers different possible outcomes, incorporating a variety of decisions. Since the order of decision nodes vary for different random seed, we can get many decision tree models with random initialization and decision processes.

D. Ensemble Methods

The paper experiments 3 different ensemble methods: Random Forest, Bagging and Adaboost. The definition of ensemble can be simplified as combing multiple classifier predictions and vote for the final prediction. Ideally, ensemble techniques can improve the performance of the dataset dramatically even when most of the classifiers get poor accuracy or metrics.

1) Bagging: Bagging takes consideration of the instances individually, taking the one with the majority of votes as the selected prediction.

2) Random Forest: The Random Forest combines different decision trees to form a *forest* which is a special kind of Bagging usually designed for classification problems. Actually, Random forest is better than bagging because it decorrelates the trees with the introduction of splitting on a random subset of features, which means that the model considers only a small subset of features at each split of the tree rather than all of the features of the model.

3) Adaboost: this ensemble method adjusts weights for different classifiers, increasing weights for those data incorrectly classified and doing the opposite for correctly classified data step by step. Unlike Bagging, Adaboost can not be done in

	acc (%)	precision (%)
Naive Bayes	19	25.8
Naive Bayes TFIDF	23.25	24.4
Logistic Regression TFIDF	27.75	27.9

TABLE II: Baseline classifiers

parallel on GPU, so it's much slower than Bagging, but usually performs better.

E. Neutral Network

This paper utilizes two kinds of the LSTM models (single and bidirectional) to classify the Twitter message sentiments. Figure 1 shows the structure of both LSTM and BLSTM networks which demonstrates that the BLSTM inserts a additional backward layer into the LSTM network using the embedding with the opposite order.

IV. Experiments

A. Data Source and preprocessing

The dataset is collected by Aman Miglani posted on Kaggle [8]. There are about 45000 Tweets from different users around the world and with 5 labels: extremly positive, positive, neutral, negative and extremely negative. If we tokenize the messages by 1-gram, there are about 93000 tokens which is almost the numbers of the whole English vocabulary. So we can assume that the dataset is large enough to generate convincing experiment results.

The basic clean for the text messages include lower case, punctuation deletions, ascii code transformation etc. This step can give us clean text messages for us to tokenize, otherwise there will be many useless and meaningless tokens (usually appears only once).

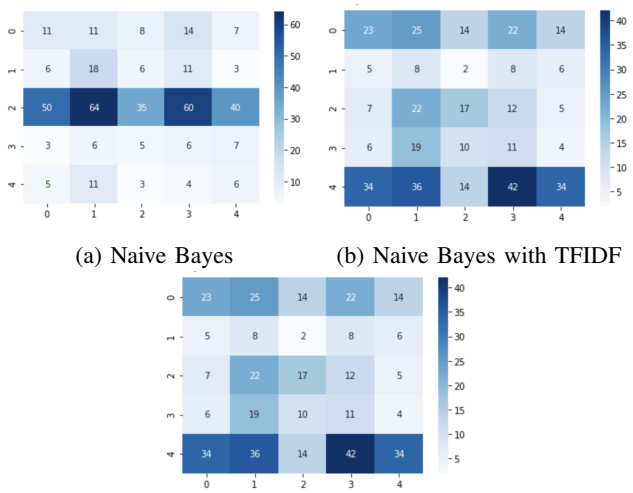
Another thing is that the five labels are mapped to integers from 0 to 4 in order to feed to the Pytorch tensor.

After the basic clean, we can tokenize the messages and transform it into a matrix using bag of words algorithm provided by Sklearn. This bag of words matrix contains 40000 rows (samples) and 90000 columns (tokens), so it's quite time consuming to train the bag of words matrix or feed it into the models. Also, we need to normalize the matrix properly and weight each value in the matrix by TFIDF (a statistical measure that evaluates how relevant a word is to a document in a collection of documents) And then, the algorithm extracts the features from the original Tweet text by PCA. In this experiment, we extract 50 features from the bag of words matrix.

B. Baseline Classifiers

This paper test different probabilistic classifiers including Naive Bayes, Logistic Regression, SVM, KNN and Decision Tree. And we will go through all these classifiers.

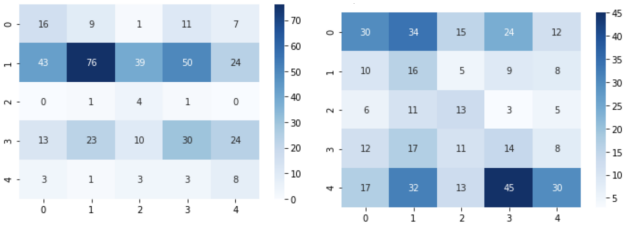
First, let's check our baseline classifiers: Naive Bayes and Logistic Regression. The table II shows that our TFIDF normalization can improve both accuracy and precision on the test dataset. However, both models perform bad on the test dataset with less than 30% accuracy.



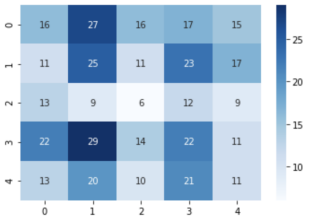
(a) Naive Bayes (b) Naive Bayes with TFIDF

(c) Logistic Regression TFIDF

Fig. 2: Baseline classifiers



(a) SVM (b) KNN



(c) Decision Tree

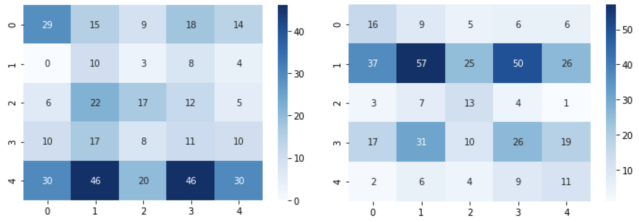
Fig. 3: Probabilistic classifiers

C. Probabilistic Classifiers

The SVM achieves the best result among all the probabilistic classifiers with about 40% accuracy. However, the Decision Tree classifier performs the worst among all probabilistic classifiers. This may due to the bad initialization for the subtree separation and random forest can address this kind of issues.

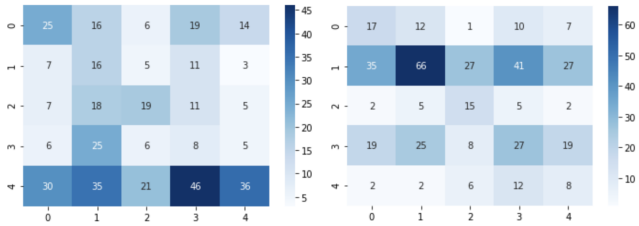
	acc (%)	precision (%)
SVM	33.5	39.5
KNN	25.75	27.7
Decision Tree	20.0	20.6

TABLE III: Probabilistic Classifiers



(a) Naive Bayes L1

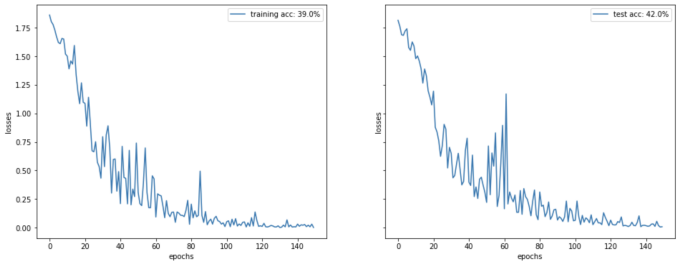
(b) Logistic Regression L1



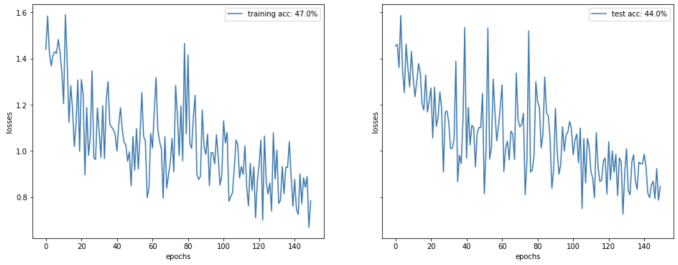
(c) Naive Bayes L2

(d) Logistic Regression L2

Fig. 4: Probabilistic classifiers with sentiment features



(a) LSTM



(b) BLSTM

Fig. 5: Neutral networks

D. Classifiers with sentiment features

Besides the knowledge-based classification, the sentence sentiment can also be viewed as new features. The key is how to formulate the sentence sentiment based on words sentiment values. Here, I define 2 ways of sentence sentiment calculation: L1 and L2

$$S_{l1} = \sum_{word \in N} word.pos - word.neg \tag{1}$$

$$S_{l2} = \sum_{word \in N} (word.pos)^2 - (word.neg)^2 \tag{2}$$

	acc (%)	precision (%)
Naive Bayes L1	24.25	28.7
Naive Bayes L2	26.0	28.0
Logistic Regression L1	30.75	33.2
Logistic Regression L2	33.25	34.2

TABLE IV: Sentiment features Classification

If we compare the Naive Bayes and Logistic Regression classifiers with or without the sentiment features, both the test accuracy and precision improves a little bit. This is good because there are 50 features extracted by PCA which concatenate with the only one sentiment feature (either L1 or L2). So the test accuracy and precision can't improve a lot with just one more good features.

E. Neutral Network

In the experiment, there are 2 neutral networks: LSTM and BLSTM. The structure of both models are introduced in previous section.

We can notice that the total epochs for both training and test are 150 and they share the same embedding input. However, the test accuracy of BLSTM is lower than LSTM by 4%. It's reasonable because I ran the code with different initialization and sometimes BLSTM can performs better than LSTM; sometimes it can not.

Meanwhile, the losses for BLSTM is not stableups and downs even after 100 epochs. One possible reason is that the BLSM classifier is much more complicated, so it may requires smaller learning rate, but we use the same hyparameters in this experiment.

F. Ensemble Methods

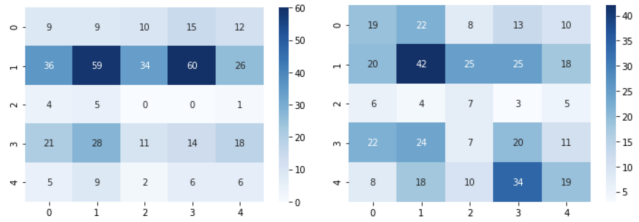
The table V shows that Random Forest improves a bit than the Decision Tree Classifier. The Random Forest performs the worst among all ensemble methods, maybe due to the poor result of decision tree classifiers. The Bagging and Adaboost don't improve the accuracy and precision much. This maybe due to that we only combine different kinds of classifiers together with only one for each. So the output maybe better if more independent classifiers are combined to do the Bagging or Adaboost.

	acc (%)	precision (%)
Random Forest	22.0	17.6
Bagging	26.75	26.8
Adaboost	26.0	25.6

TABLE V: Ensemble Methods

V. Conclusion

In this paper, we explore different kinds of probabilistic classifiers such as Naive Bayes, Logistic Regression, SVM, KNN, Decision Tree and so on. Among them, Naive Bayes and Logistic Regression are used as the baseline classifiers. We can see that the test accuracy and precision varies on other classifiers with more hyperparameters as long as we use different initializations.



(a) Random Forest (b) Bagging

(c) Adaboost

Fig. 6: Ensemble Methods

Besides the knowledge-based classification methods, we also utilize the LSTM (single and bidirectional) neutral networks to train with words order information.

Meanwhile, we combines the bag of words with the sentence sentiment features, in which two kinds of words sentiment calculation methods are proposed.

Finally, there are 3 different ensemble techniques explored, which combines and improves multiple classifiers and meant to get good result even we only have some classifiers with poor performances.

REFERENCES

[1] A. Esuli and F. Sebastiani, "Sentiwordnet: A publicly available lexical resource for opinion mining," 05 2006.

[2] C. Akkaya, J. Wiebe, and R. Mihalcea, "Subjectivity word sense disambiguation," 01 2009, pp. 190–199.

[3] G. Liu and J. Guo, "Bidirectional lstm with attention mechanism and convolutional layer for text classification," *Neurocomputing*, vol. 337, pp. 325 – 338, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231219301067

[4] R. Chen, Y. Zhou, L. Zhang, and X. Duan, "Word-level sentiment analysis with reinforcement learning," *IOP Conference Series: Materials Science and Engineering*, vol. 490, p. 062063, 04 2019.

[5] F. Huang, K. Wei, J. Weng, and Z. Li, "Attention-based modality-gated networks for image-text sentiment analysis," vol. 16, no. 3, 2020.

[6] E. Cambria, B. Schuller, Y. Xia, and C. Havasi, "New avenues in opinion mining and sentiment analysis," *IEEE Intelligent Systems*, vol. 28, no. 2, pp. 15–21, 2013.

[7] G. Liu and J. Guo, "Bidirectional lstm with attention mechanism and convolutional layer for text classification," *Neurocomputing*, vol. 337, pp. 325 – 338, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231219301067

[8] A. Miglani, "Coronavirus tweets nlp - text classification corona virus tagged data," 2020.