Sentiment Analysis over Twitter Data *

Yun Wu **Qiren Chen** **Xiaofan Lu**
University of Texas at Austin
Computer Science Department
{ywu, chenqr, xiaofan}@cs.utexas.edu

Abstract

In this project, we address the issue of sentiment analysis of twitter messages. We collect three corpora from different domains and propose several pre-processing steps based on the nature of the corpus. Features based model and Recursive Neural Tensor Network (RNTN) are evaluated by extensive experiments. Experiment results indicate that RNTN model works better on well formatted text and feature-based model performs better on general topic tweet. To the best of our knowledge, this is the first work on twitter sentiment analysis which accounts sentence structure.

1 Introduction

Microblogging websites (such as Twitter, Weibo) have gained popularity in recent years. People can easily post real time, short messages to express their opinions. Sentiment of microblog is of particular interest because such information is valuable in diverse areas such as entertainment, politics and economics.

However, sentiment analysis over twitter message is a challenging task due to the noisy nature of tweet. It contains ungrammatical sentences, typos, creative punctuations, slangs, new words, URLs, and genre-specific terminology and abbreviations, such as, RT for re-tweet, #hashtags, @mentions.

Most of existing works of sentiment analysis are based on bag of words classifiers. Different features to improve the performance of bag-of-words model have been proposed (Agarwal et al., 2011). Such classifiers can work well in longer documents by relying on a few words with strong sentiment such as ”great” or ”awesome”. However, bag-of-words models have difficulties in handling negations and comparisons, which rely on the structure of sentence.

Recursive Neural Tensor Network (RNTN) model is recently proposed to capture the compositional effects with higher accuracy. It represents a phrases through word vectors and a parse tree and then computes vectors for higher nodes in the tree using the same tensor-based composition function. (Socher et al., 2013). The authors claim that RNTN can accurately captures the sentiment change and scope of negation. However, the data set used in the above paper is a bunch of single sentences extracted from well formatted movie reviews, which is quite different from twitter message.

In this paper, we evaluate both feature-based and RNTN models on twitter message. We collect twitter message from different sources and labeled some of them. Such corpus has been pre-processed according to the natural of different models. Experiment results indicate that RNTN perform significantly better than SVM on well formatted movie reviews. When training on *Sentiment Treebank* and testing on general topic tweet, RNTN still performs slightly better. However, due the lack the annotated training set on twitter message, existing RNTN model fails outperform in SVM on the task of sentiment detecting on general topic tweet. In the end, we explore the possibility of domain adaptation over RNTN model.

*This is the final project report for CS 388 Natural Language Processing at The University of Texas at Austin in 2014 Spring

2 Problem Definition and Algorithm

2.1 Task Definition

In this paper, we address the problem of sentiment analysis of Twitter message. To be more precise, given a message, we want to classify whether the message is of positive or negative (binary decision), or neutral sentiment (ternary decision). For the following two example messages, we are expected to return positive on the first one and negative on the second.

> If u haven't seen #Rio2 yet-GO! You need to meet Gabi! Great singer. Cute. Absolutely hysterical! @KChenoweth `pic.twitter.com/kkVBUKjqE3`

> The rio 2 has one of the worst soundtracks evvvvaaa. I'm at Alamo @Drafthouse Cinema. @marissanicole11 `http://4sq.com/1kKF8qE`

This is an interesting task because sentiment of Twitter message can be used as a barometer for public mood and opinion in diverse areas such as entertainment, politics and economics. For example, it was used to provide information on the temporal dynamic of sentiment in reaction to the debate video between Barack Obama and John McCain(Diakopoulos and Shamma, 2010). There is also a report indicates that sentiment toward public figure may have potential influence over stock market.[1]

However, twitter message presents greater challenges for sentiment analysis than traditional text genres, such as newswire data. Tweets are within 140 characters, often consists of a few short sentences or even a single sentence. The language used is very informal, with creative spelling and punctuation, misspellings, slang, new words, URLs, and genre-specific terminology and abbreviations, such as, RT for re-tweet and #hashtags, which are a type of tagging for Twitter messages. [2]

2.2 Algorithm Definition

We experiment with two different types of models: The first one is feature based model. In this model, we use different combination of features and SVM classifier. The other is the recently proposed Recursive Neural Tensor Network which works pretty good on movie reviews. We first introduce the pre-processing steps and then the models.

2.2.1 Pre-processing

The Twitter language is known as informal and flexible. Such properties will provide information about sentiment and should be processed carefully. As the feature and RNTN models are built upon two different basic ideas, we need to have different pre-processing steps for each model. For the feature based model, our goal is to maximize the information about sentiment while reduce the sparsity of the feature vector. For RNTN, we try to formalize tweets to well-organized sentences such that Stanford parser can recognize it well.

General features First we replace all the remaining HTML entities like "&amp;" to ASCII code "-". We also delete the quotes around the whole tweet. Then we transform all the URL addresses to the keyword "URL" and the user names(start with @) to the key word "target" to keep the structure of the sentence. Hashtags may contain sentimental information so we simply remove the "#" in the front of the hashtag. We transform all the lengthening words ("sooooo" instead of "so") to three repeated letters like "sooo". There are also many misspellings and slangs in Twitter and we use a slang dictionary to formalize it.

Word Cluster We also try to reduce feature space with word cluster (Owoputi et al., 2011). Firstly, we obtain hierarchical word clusters via Brown clustering. The algorithm partitions words into a base set of 1,000 clusters, and induces a hierarchy among those 1,000 clusters with a series of greedy agglomerative merges that heuristically optimize the likelihood of a hidden Markov model with a one-class-per-lexical-type constraint. Then we sort the words in each cluster with frequency and use the most frequent one to represent the sentiment of the cluster. According to our experiment, word cluster does help with nonstandard expressions:

[1] `http://goo.gl/W1fY1c/`

[2] `http://www.cs.york.ac.uk/semeval-2013/task2/`

{001010110} never neva nvr nevr #never neverr nver neverrr nevaa neeever nevet

It also integrates antonyms:

{1111000100011} sad blessed frightening lucky frustrating helpful impressed :(((

Preprocessing for SVM We extract additional features for SVM and attach "_neg" to negated sentences. Detailed information can be found in Section 2.2.2.

Preprocessing for RNTN Since RNTN is based on Stanford parser, we need to preserve the structure of sentences while filtering out noise. Emoticons, for example, can not be easily organized by our parser but provide sentimental information. We replace each emoticon with the corresponding adverb. For instance, ":-)" will be replaced with "happily", ":O" with be substituted as "surprisedly". Also, to reduce errors in the sentence splitter, a period is added to each tweet that does not end with a punctuation.

2.2.2 SVM

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. We use the SVM$^{\text{libSVM}}$ software with a linear and Radial Basis Function (RBF) kernel. We input two sets of vectors of size m. Each entry in the vector corresponds to the presence of a feature. For example, with a unigram feature extractor, each feature is a single word found in a tweet. If this single word also appears in our dictionary, the value is 1, otherwise the value is 0. In our experiment, we evaluate the following features:

1. **Word unigram and bigram** (Unigram or Bigram): We use three different methods to present unigram feature.
   Bianry Format: We use $0/1$ to indicate the presence of a word.
   Frequency: The count of each word is used as unigram feature. For example, if good appears twice in the Tweet, we set 2 to this word.
   tf-idf: It is a numerical statistic to reflect the importance of a word to a document. tf-idf is the product of two statistics, term frequency and inverse document frequency.

$$tfidf_{i,j} = tf_{i,j} \times idf_i \quad (1)$$
$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \quad (2)$$
$$idf_i = \log \frac{|D|}{|\{j : t_i \in d_j\}|} \quad (3)$$

2. **Elongated word** (-elongated): Lengthening words by repeating letters (such as coooool, goooood) is reported to be a strong indication of sentiment. Brody et al. (2011) reported that on average, one out of six sentences has word lengthening. In our experiment, we also take elongated word as a feature.

3. **Targets, Hashtags, URLs, Numbers** (-url-num): These are twitter specific features. Users of Twitter use the "@" symbol to refer to other users and use hashtags # to mark topics.

4. **Negation** (neg): The effect of negation word is not simply the opposite of the original sentiment. In this paper, we try to append an NEG suffix to every word appearing between a negation and a clause-level punctuation mark.

2.2.3 RNTN

In Recursive Neural Tensor Network, each word is represented as a $d-$dimensional vector. When an $n-$gram is given to the compositional models, it is parsed into a binary tree (as in Figure 1). We compute the parent vector in a bottom up fashion using a compositionally function g and use node vectors as features for a classifier at that node.

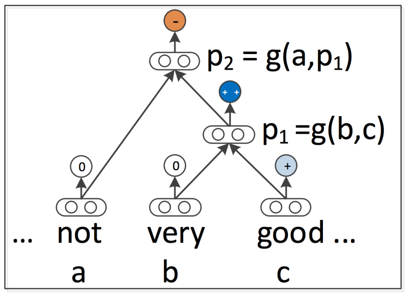


Figure 1: Trigram Example (Socher et al., 2013)

RNTNs use the following equations to compute the parent vectors:

$$p_1 = f\left(\begin{bmatrix} b \\ c \end{bmatrix}^T V^{[1:d]} \begin{bmatrix} b \\ c \end{bmatrix} + W \begin{bmatrix} b \\ c \end{bmatrix}\right)$$

where $f = \tanh$ is standard element-wise non-linearity. $V^{[1:d] \in \mathbb{R}^{2d \times 2d \times d}}$ is the tensor that defines mulitiple bilinear forms. $W \in \mathbb{R}^{d \times 2d}$ is the main parameter to learn.

The next parent vector p_2 in the tri-gram will be computed with the same weights:

$$p_2 = f\left(\begin{bmatrix} a \\ p_1 \end{bmatrix}^T V^{[1:d]} \begin{bmatrix} a \\ p_1 \end{bmatrix} + W \begin{bmatrix} a \\ p_1 \end{bmatrix}\right)$$

As we use the RNTN model as a black box in this project, so I skip the details on how to train the model. Interested reader could refer to the paper by Socher et al,. (2013).

3 Experiment

3.1 Corpus

We conduct our experiments on three corpus.

- The first corpus is the Stanford sentiment treebank released by Socher et. al. (2013). It is based on the dataset introduced by Pang and Lee (2005) and consists of 11,855 single sentences extracted from movie reviews on `http://www.rottentomatoes.com/`. It was parsed with the Stanford parser (Klein and Mannning, 2003) and includes a total of 215,154 unique phrases from those parse trees, each annotated by 3 human judges. To the best of our knowledge, it is the only public available corpus upon which a RNTN sentiment model can be trained right now. We refer to this corpus by *Sentiment Treebank* in the rest of this paper.

- The second corpus is movie reviews on Twitter. We have 364 tweets extracted from two specialized review accounts (@FilmReviewIn140, @MovieTwoosh). Such reviews are mostly well formatted, usually consisting of several sentences. We label the tweet based on the A to F grades rated by author. Tweets with a grade no worse than B are labeled positive otherwise negative. We refer to this corpus by *moive* in the rest of this paper.

- The third corpus is general tweet messages. It is taken from SemEval-2013: Sentiment Analysis in Twitter Task B[3]. In their release, each of the tweet messages has been manually labeled as positive, negative, or neutral. Out of all the 5,750 messages, 2,042 are positive, 855 are negative and 2853 are neutral. We refer to this corpus by *SemEval* in the rest of this paper.

3.2 Single Sentence Sentiment

We firstly evaluate both models using *Sentiment Treebank*, The same training/testing split as in the original paper(Socher et al., 2013) is used.

Model	Accuracy		
	positive	negative	overall
RNTN	80.83	87.91	84.27
SVM$_S$	74.15	73.79	73.97
SVM$_L$	75.90	73.90	74.90

Table 1: Binary decision

In SVM$_S$, we use a smaller dictionary (1635 words) in which words appear more than 10 times are used. In SVM$_L$, we build a larger dictionary (3504 words) with the bar lowered to 5 times.

3.3 Multiple Sentences Sentiment

We then evaluate how to combine the sentiment of multiple sentences. This is not an issue of feature-based model because only a larger bag is needed. However, RNTN relies on the structure of single sentence so we need to combine the sentiment from multiple sentences within a single tweet. Here, we train the model on *Sentiment Treebank* and test it on *movie* corpus.

As for the RNTN model, we evaluate two ways to combine the sentiment of the whole tweet (multiple sentences). The first is to make hard (binary) decision on single sentence (either positive or negative) and sentiment of the tweet is decided by majority vote. Soft information (probability) is only used to

[3]`http://www.cs.york.ac.uk/semeval-2013/task2/index.php?id=data`

break a tie. The second way fully relis on soft (probability) information. For each sentence, we generate a 5-element vector for the probability of the having the corresponding sentiment (very negative, negative, neutral, positive, very positive). We add the vector for all the sentences together and make final decision based on the combined vector.

Model	Accuracy(%)		
	positive	negative	overall
RNTN$_{hard}$	70.08	81.54	74.18
RNTN$_{soft}$	78.21	80.0	78.85
SVM	73.50	66.92	71.15

Table 2: Binary decision

For the two RNTN models, hard decision model has worse performance than soft decision in positive sentiment but slightly better on negative sentiment. This is reasonable because more information is available in soft decision combining. Based on the above result, we use soft mode in the following experiments.

3.4 General topic Tweet

We conducted three sets of experiment over the general topic twitter corpus *SemEval*.

- Exp 1: Training on 90% of *SemEval* and testing on the rest 10%. Different feature for SVM are evaluated

Feature		Accuracy (%)
	Binary Format	78.84
Unigram	Frequency	78.64
	Tf-idf	78.22
Bigram		70.41
Unigram-Bigram		78.26
Unigram-url-num		**79.32**
Unigram-elongated		79.01
Unigram-url-num-elongated		79.19
Unigram-neg		78.77
Unigram-neg-url-num		78.19
Unigram-neg-enlongated		78.34
Unigram-neg-url-num-enlongated		77.95

Table 3: Experiment 1

This is the classic experiment setup. However, as we don't have annotated sentiment parse trees (the sentiment of each node of the parse tree, which are required to train the RNTN model), a RNTN model based on *SemEval* can not be trained. Thus, we evaluate the following feature-based models in this experiment.

- Exp 2: Training on *Sentiment Treebank* and testing on *SemEval*.
  In this experiment, we apply the model trained on *Sentiment Treebank* to the general topic twitter message. It may sound wierd but as we mentioned earlier, *Sentiment Treebank* is the only corpus upon which we can train RNTN model. To be fair, we train a feature-based model in the same way and compare the performance of the two in both binary-decision and ternary-decision task.

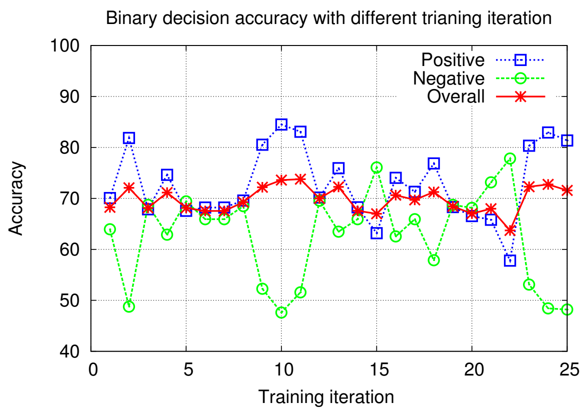
Model	Accuracy(%)		
	positive	negative	overall
RNTN	69.83	70.17	69.93
SVM	67.14	60.91	65.30

Table 4: Experiment 2.1

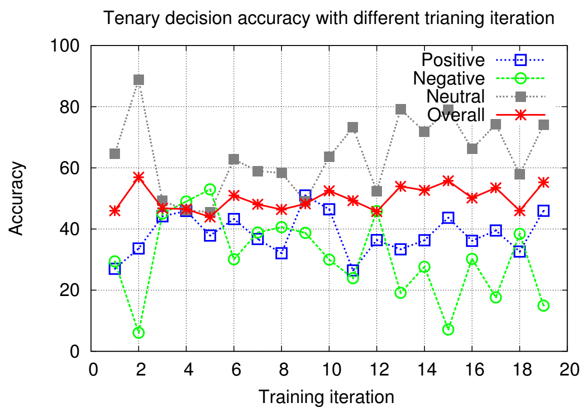
Model	Accuracy(%)			
	positive	negative	neutral	overall
RNTN	48.12	43.17	49.40	48.02
SVM	52.52	23.86	44.69	37.14

Table 5: Experiment 2.2

- Exp 3: Domain adaptation of RNTN
  In this experiment, we explore domain adaptation of RNTN sentiment model. This is an interesting experiment as gathering fully annotated training data on twitter message is expensive and labor intensive. We have fully annotated corpus in *Sentiment Treebank* but *SemEval* is only been labeled positive, neutral or negative. In this experiment, we first train the model on *Sentiment Treebank*, then use the above model to label the training set of *SemEval* in sentiment treebank format. We re-



(a) Binary Decision



(b) Ternary decision

Figure 2: Domain adaptation of RNTN

train the model with both *Sentiment Treebank* and the labeled training set of *SemEval*. In the end, we test the new model on the test set of *SemEval*.

We train the model with different iterations and evaluate all the models. As we can see from Figure 2, the result fluctuates a lot. Unfortunately, we don't see significant improvement in performance with domain adaptation.

3.5 Discussion

3.5.1 Feature analysis

- **Unigram**: Experiment results indicate that unigram feature in binary format performs best. The result of using frequency and Tf-idf to present unigram is slightly worse than using binary format. This is because Tweet is rather short (at most 140 characters). For this reason, frequency is not as a good feature in Tweets as in other corpus. For Tf-idf, also because there are so many topics in Tweets, each word is hardly used to distinguish sentiment.

- **Bigram**: We use bigrams to capture negated phrases like "not good" or "not cool". In our experiments, Bigram as a feature fails to improve accuracy. This is because bigrams tend to be very sparse and the size of our corpus is relatively small. In general using bigrams as features is not helpful because the feature space is very sparse. Thus we want to combine Unigram and Bigram as features.

- **Unigram-Bigram**: Results show Unigram-Bigram as a feature doesnt help. Compared to Unigram features, accuracy drops from 78.84% to 78.26%. For Alec Go (2009), there was also a decline for Unigram-Bigram as features.

- **Targets, Hashtags, URLs, Numbers**: The feature combining URL, Numbers with Unigram performs best in our experiment. The result shows that the number of URLs and Numbers is helpful for this task.

- **Elongated**: Elongated word as a feature doesnt help. The reason is that we can hardly classify different elongated forms of one word into one classification, such as goood and gooooddddd. So elongated words as a feature are sparse and hardly provide other information. In the future work, if we have a better algorithm to deal with elongated words, we can improve the results.

- **Negations**: Negations as features work almost as well as Unigram as a feature. The reason why negations dont improve the results is that it makes the dictionary larger than the original one. But since the size of corpus is relatively small, so negations dont show its advantage in our experiment.

3.5.2 Comparison of SVM and RNTN

As we can see from Table 1, the performance of RNTN model is significantly better than SVM models on well formatted text. This is consistent with the result of the original paper (Socher et al., 2013). The huge boost comes from the fact that the structure of the sentence is utilized in the RNTN model.

From Table 4 and Table 5, we can see that RNTN out perform SVM in both binary decision and ternary decision. However, the performance of SVM is worse than the result from Table 3. The huge decrease in SVM model results from the difference of the two corpora. Clearly, twitter specific feature helps SVM model a lot but such feature is not available in the *Sentiment Treebank*.

Another interesting comparison is the performance of RNTN in Table 1 and Table 5. Like statistical parser, RNTN model is also quite specific to the genre of the training corpus. To make the problem worse, the Stanford parse doesn't work quite well on twitter message due to the noisy nature of it.

4 Related Work

Most of previous work on sentiment analysis is based on bag-of-word model. Alec Go provided a method to use n-gram as features to analyze sentiment of Twitter (Go et al., 2009). A. Agarwal introduced POS-specific prior polarity features and tree kernel (Agarwal et al., 2011). S. Mohammad combined several different features extracted from each Tweet (Mohammad et al., 2013). None the above work take the structure of sentence into account.

As we mentioned before, RNTN model has been tested over movie reviews and significantly outperform the bag-of-word models(Socher et al., 2013). However, it has not been tested on twitter message yet. In our project, we apply RNTN model over Twitter corpus and compare its performance with SVM model.

5 Future work

- Better pre-processing technique
  Due to the noisy nature of twitter messages, how well we can preprocess it matters a lot. How to preserve the structure of the original sentence and filter out unnecessary information is a challenging task.
- Better parser for twitter message
  As the RNTN is built upon parse trees, the performance of the parser is of great importance to RNTN model. However, most of existing parsers, including the one we used (Stanford Parser) are trained on newswire data, which is very different from the twitter messages. We believe within a few years, the focus of the research community on twitter message will shift from current POS tagging to building an accurate parser.
- Sentiment treebank over twitter message
  Our experiments are limited by the fact that there is no labeled sentiment treebank of twitter messages. This is largely because that RNTN model has just come out. Building our own sentiment treebank would be too much for us as a course project. However, it would be of great help if such a sentiment treebank over twitter messages is available to the research community.

6 Conclusion

In this project, we evaluate both feature-based and RNTN models on the task of twitter sentiment analysis. With the same setup, RNTN model performs better than feature-based model. However, due to the lack of the annotated corpus (parse tree with sentiment label) over twitter messages, RNTN model has worse performance than feature-based model on sentiment decision over general topic tweets. Domain adaptation of RNTN model from movie reviews to twitter messages gives no significant improvement.

Acknowledgments

We would like to thank Dr. Mooney and our TA Zihao Zhou for their guidance and help on this course project.

References

[Socher et al.2013] , Socher, Richard and Perelygin, Alex and Wu, Jean and Chuang, Jason and Manning, Christopher D. and Ng, Andrew Y. and Potts, Christopher. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebanka *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*

[Diakopoulos and Shamma 2010] , N. Diakopoulos, D. A. Shamma. April, 2010. Characterizing Debate Performance via Aggregated Twitter Sentiment. *Conference on Human Factors in Computing Systems (CHI).*

[Agarwal et al.2011] , Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. 2011. Sentiment analysis of Twitter data. *In Proceedings of the Workshop on Languages in Social Media (LSM '11). Association for Computational Linguistics, Stroudsburg, PA, USA, 30-38.*

[Owoputi et al.2011] , Olutobi Owoputi and Chris Dyer and Kevin Gimpel and Nathan Schneider and Noah A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters *In Proceedings of NAACL*

[Hu et al.2004] , Hu M, Liu B. 2004. Mining and summarizing customer reviews *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2004: 168-177.*

[Go et al.2009] , Go A, Bhayani R, Huang L. 2009. Twitter sentiment classification using distant supervision *CS224N Project Report, Stanford, 2009: 1-12.*

[Mohammad et al.2013] , Go A, Bhayani R, Huang L. 2013. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets *arXiv preprint arXiv:1308.6242, 2013.*

[Das and Chen2001] , Sanjiv Das and Mike Chen. 2001. Yahoo! for Amazon: extracting market sentiment from stock message boards. *In Proceedings of the 8th Asia Pacific Finance Association Annual Conference.*

[Bo et al.2002] , Pang, Bo; Lillian Lee; and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. *In Proceedings of the Conference on Empirical Methods in Natural Language Processing. ACL.*