LING 406 Final Report

Yuchen Liang

yliang35@illinois.edu

May 12, 2022

This report aims for extra credits.

1 Introduction

Sentiment analysis aims to determine the sentiment given a piece of text. Despite diverging definitions for the word *sentiment* (Mohammad, 2020), in this report the meaning is restricted to a binary polarity in emotion. Emotional polarity, though limited compared to the entire range of human emotions, is practically useful for many applications, including those in economy, finance, sociology, and politics, where it serves as a strong factor in the decision-making process (such as financial forecasting).

2 Problem Definition

The task of sentiment analysis can be further divided into three categories: document-level, sentence-level, and aspect-level. In this report, sentence-level sentiment analysis is of interest. The task for the system is, given a piece of text (usually a review or tweet), to label the sentence with carrying either positive or negative sentiment. To develop the system, it receives a corpus of fully labeled sentences (e.g., movie reviews, restaurant reviews, tweets, etc.), where the label is the sentiment of every sentence.

3 Previous (Recent) Work

Traditional rule-based methods majorly focus on document-level and sentence-level sentiment analysis (Wang and Manning, 2012; Nakov et al., 2013), but after 2014, aspect-based sentiment analysis (ABSA) becomes more popular, in that the task is more fine-grained and potentially

more helpful to the industry. The task of ABSA is similar, but the difference is that the system needs to output the sentiment regarding a specific given aspect described (e.g., the food or service of a restaurant). The output could be one of positive, negative, neutral, and conflict, the last of which means that two conflicting sentiments are expressed in a single sentence. Moreover, many researchers are trying to tackle this problem from a deep learning perspective, due to recent introduction of computation power and advances in deep learning fields. In the following, three different types of deep learning models are identified.

The first type of methods to introduce uses Convolutional Neural Network (CNN) for ABSA (Mulyo and Widyantoro, 2018). Given each input text, the authors model it as an 100 * 100 image, where the first number represents the maximum number of words in a sentence and the second number represents the dimension of the GloVe (Pennington et al., 2014) embedding for each single word. If the text has length less than 100, zeros are padded to the end of the text. Then, ABSA is converted to an image classification problem, drawing usual techniques from the Computer Vision community. The SemEval-2015 dataset is used, and the results are compared to other ML algorithms that they implemented. However, drawbacks of this approach include that one needs an upper threshold for the sentence length, and that sequential properties are hard to capture using CNN.

The second type of methods uses Recurrent Neural Network (RNN), and more specifically Long Short-term Memory (LSTM) unit, for the ABSA task (Tang et al., 2015, 2016; Wang et al., 2016; Ma et al., 2017). The first LSTM to solve sentiment analysis task appears in 2015, which is at the document-level (Tang et al., 2015). The idea is to learn a hidden sentence representation using LSTM. The input to the LSTM is the GloVe word embedding of dimension 300. The final hidden state (or an average of all hidden states, where both implementations are discussed in the paper) is fed into an output layer for sentiment classification. Soon after that, to solve ABSA, the same authors proposed LSTM methods that incorporate aspect-level information into the system by training two LSTMs of opposite directions that both stop at the aspect words (Tang et al., 2016). Others proposed using basic and sophisticated attention mechanisms to focus more on important words (Wang et al., 2016; Ma et al., 2017). Their datasets include SemEval datasets from 2014 to 2016 (or their mixture), and the performance is getting better over the years, which

recently slightly outperforms the best known traditional method that uses SVM (Kiritchenko et al., 2014).

The third type of methods uses Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) to solve the ABSA task (Hoang et al., 2019). The authors model the task as a semantic similarity task. Specifically, one application of BERT is next-sentence prediction, where two sentences separated by [SEP] are classified for continuity (e.g., whether or not the latter is the next sentence of the former, etc.) based on semantic similarity. For the ABSA task, the authors fine-tune the pre-trained BERT model, and they regard the first sentence as the given text and the second sentence as the tested entity plus aspect. The performance is measured on SemEval-2016 dataset.

4 Approach

4.1 Datasets

The first dataset in the experiment is the movie review dataset (Pang and Lee, 2004). It contains 64720 sentences of movie reviews. Originally in star scale, the reviews are further categorized into positive and negative sentiments. In the experiment, the dataset is divided randomly into the training (60%), validation (20%), and testing set (20%).

The second dataset is the Yelp review dataset in the Champaign-Urbana area, prepared by a former Linguistic student John Hall. There are 10391 reviews in the dataset on a 5-star scale. Among them, those reviews with 4 and 5 stars are labelled as positive (6066 reviews), and the rest are labelled as negative (4325 reviews). It is then similarly divided randomly into the training, validation, and testing set.

4.2 Baseline System

In the baseline system, the bag-of-words feature is used to tackle the problem. The dataset from Pang and Lee is already normalized and tokenized, so no text pre-processing is done at this stage. For the Yelp dataset, different paragraphs in the single review are concatenated, and the NLTK $word\_tokenizer$ is run on the entire long string.

For each word, the number of occurrence in the positive and negative corpus is counted respectively, and the log-probability of the word being in a positive context is calculated. To

handle unseen words, an additional one count, representing $\langle \texttt{UNK} \rangle$, is also added to both the positive and negative corpus. During testing, the log-probability of each word in the given text is calculated. The average of them is compared to $\log 0.5$, and the text is labelled as positive if it crosses this threshold.

In the following, several feature improvements are mentioned, which, if put together, becomes the overall improved system.

4.3 Text Pre-processing: Negation Handling

Negation handling is the first major improvement to the system. As a first step, all abbreviations of "n't" are casted into regular "not" (following a white space) during the text pre-processing stage. Then, keywords such as "not", "never" are identified as the start of negated context, which ends either at the punctuation mark or before keywords signaling subordinate sentence (e.g., "that", "where", etc.). Following the NRC Lexicon, only adjectives, adverbs, verbs, and nouns are identified, and the `_NEG` flag is attached to the original word.

4.4 Number of Positive/Negative Tokens

The second improvement is to add the number of positive and negative words in the classification. This is helpful because repetition (of synonyms) in a given text is usually used to show the speaker's sentiment. This feature is calculated for both log-probabilites and PMI scores (defined below).

4.5 N-gram

The third improvement is to use N-gram, specifically bi-gram, in the system. The log-probabilities and Point-wise Mutual Information (PMI) scores are calculated for both unigrams and bigrams. Following previous practices (Mohammad et al., 2013; Kiritchenko et al., 2014), for each token w (either a unigram or a bigram-pair), the PMI score is defined as

$$score(w) := PMI(w, pos) - PMI(w, neg)$$

where

$$PMI(w, l) := \log_2 \frac{freq(w, l) * N}{freq(w) * freq(l)}, \ \forall l \in \{pos, neg\}.$$

Here $freq(w, l)$ is the number of times w occurs in corpus with sentiment l, $freq(w)$ is the total number of occurrence of w, $freq(l)$ is the total number of tokens in the corpus with sentiment l, and N is the total number of tokens in the entire corpus. Since PMI is a poor estimate for low-frequency words, only those words with occurrence more than 10 is considered for PMI scores.

4.6 Part-of-Speech Analysis

The fourth improvement is to incorporate part-of-speech (POS) information in the system. This is done using the built-in function in NLTK. Words of particular interest are those adjectives, adverbs, verbs, and nouns, which are helpful to sentiment classification. As mentioned in text pre-processing, POS is used in negation labeling, and it is also used to obtain extra features for a given sentence, where the log-probabilities and PMI scores of the words are accumulated for each of the four categories respectively.

4.7 External Lexicon

The last improvement is to use external lexicon. The NRC Emoticon Affirmative Context Lexicon and NRC Emoticon Negated Context Lexicon is used, which is trained on 1.6 million tweets (Mohammad et al., 2013). The PMI score is calculated and recorded for each token (i.e., either unigram or bigram). Although this lexicon is originally obtained from Twitter tweets, it might also shed some light for movie reviews.

4.8 Overall Improved System

Combining all elements above, given a piece of text, the improved system takes the following group of features for sentiment classification:

1. Number of positive and negative words
2. Sum, max, and min of sentiment scores (if available) of unigram and bigram tokens (except bigram log-probabilities, which turns out to be a bad feature). Here scores refer to both the log-probabilities and the PMI scores
3. Sum of sentiment scores (if available) from, in particular, adjectives, adverbs, verbs, and nouns, respectively

4. Sum of sentiment scores (if available) from NRC Lexicon

4.9 Machine Learning Algorithms

Four machine learning (ML) algorithms are considered in this report: SVM classifier (with RBF kernel), Gaussian Naïve Bayes classifier, Decision Tree classifier, and Neural Net classifier. Among them, the SVM classifier has been widely used for traditional sentiment classification (Wang and Manning, 2012; Kiritchenko et al., 2014). For the Neural Net, the number of hidden layer is 100, the activation function is ReLU, the initial learning rate is 0.01, and the rate is adaptive in a way that it drops to 1/5 if the training loss does not largely decrease for two consecutive epochs. All implementations of the algorithms are from Python Scikit-Learn package, and the best regularization parameters are obtained from the validation set. Their performances are recorded in Table 1.

5 Results and Ablation Study

5.1 Movie Review Dataset

Model	Accuracy	Precision (+/-)	Recall (+/-)	F1 Score (+/-)
Baseline	0.678	0.633 / **0.784**	**0.874** / 0.475	**0.734** / 0.591
SVM	**0.712**	**0.714** / 0.709	0.723 / **0.700**	0.718 / **0.704**
Naïve Bayes	0.659	0.648 / 0.673	0.722 / 0.593	0.683 / 0.630
Decision Tree	0.653	0.666 / 0.640	0.638 / 0.668	0.652 / 0.654
Neural Net	0.703	0.696 / 0.712	0.740 / 0.664	0.717 / 0.687

Table 1: Test performance of different methods on the movie review dataset. The precision, recall, and F1 score are reported for both positive and negative classes. Numbers in bold obtain the best score in their column. The evaluation dataset is the movie review dataset (Pang and Lee, 2004). All implementations come from Python Scikit-Learn package. The best SVM has $C = 0.25$, and the best Decision Tree has $max\_depth = 7$.

In Table 1, the performance of different ML algorithms on the movie review dataset is recorded. The best improved system is the one that uses SVM, which obtains nearly 3% increase in accuracy from the baseline system. This increase majorly comes from those reviews with negative sentiments, where the SVM classifier does a good job to label them.

In Table 2, an ablation study is performed on the movie review dataset for features in five categories. The models to start with are those without Negation implemented, because in the experiment, adding negation only worsens the performance for all four models. This is probably

Model	w/o Negation	+ Negation	- # Tokens	- Bigram	- POS	- NRC
SVM	0.711	0.705	**0.674**	0.705	0.710	0.710
Naïve Bayes	0.656	0.648	0.651	**0.640**	0.654	0.648
Decision Tree	0.657	0.647	0.651	0.664	**0.650**	0.657
Neural Net	0.707	0.696	**0.665**	0.707	0.710	0.704

Table 2: Ablation study of the improved methods on the movie review dataset. Accuracy is employed for the metric to compare. The evaluation dataset is the movie review dataset (Pang and Lee, 2004). The numbers in bold represent the greatest drop in accuracy for the current model in case a feature is dropped (i.e., in the row). All implementations are from Python Scikit-Learn package. The SVM parameter has $C = 0.25$. The Decision Tree has $max\_depth = 7$. The model without negation is used for baseline performance because negation has a negative effect on the overall performance.

because the data set is not large enough. If this is true, dissecting out the negated context would reduce the number of tokens and result in a less precise estimate.

For the movie review dataset, the most influential feature for SVM and NN is the number of positive and negative tokens. This feature is more helpful than the sum of scores probably because the variance of scores is relatively high compared to their mean, and thus when adding the scores together across the sentence, the noises would make the overall sum less accurate. The most influential feature for Naïve Bayes is the bigram PMI score, which also has a moderate effect on SVM and NN.[1] This is helpful probably because bigram could be useful to detect word combinations (which include negation, but without underfitting this time), and this inter-word dependency helps Naïve Bayes. The most influential feature for Decision Tree is the POS information, though the number of tokens has a similar importance. This is probably because dissecting scores for different POS help the Decision Tree to better distinguish useful scores instead of combining scores from all words into a single number.

5.2 Yelp Review Dataset

In Table 3, the performance of different ML algorithms on the U-C Yelp review dataset is recorded. From all performance metrics, the Yelp dataset is a much easier dataset for sentiment analysis. This partly due to the length of text. The length of text in the movie review dataset is usually fewer than 50 (i.e., the length of a sentence), but the average length in the Yelp review dataset reaches 100 or more (i.e., the length of a paragraph). A greater length not only provides a better score estimate but also reduces the variance in the scores as they are accumulated. The best

[1] However, bigram log-probabilities would dramatically hamper the performance. This is probably due to insufficient bigram pairs which again lead to a less precise estimate.

Model	Accuracy	Precision (+/-)	Recall (+/-)	F1 Score (+/-)
Baseline	0.848	0.843 / 0.857	0.909 / 0.763	0.875 / 0.807
SVM	0.841	0.832 / 0.859	0.913 / 0.741	0.870 / 0.796
Naïve Bayes	0.835	0.811 / **0.884**	**0.935** / 0.695	0.869 / 0.778
Decision Tree	0.844	0.848 / 0.836	0.892 / 0.777	0.869 / 0.805
Neural Net	**0.866**	**0.894** / 0.829	0.874 / **0.855**	**0.884** / **0.842**

Table 3: Test performance of different methods on C-U Yelp review dataset. The precision, recall, and F1 score are reported for both positive and negative classes. Numbers in bold obtain the best score in their respective column. All implementations come from Python Scikit-Learn package. To enforce stronger comparison, the same regularization parameters for SVM and Decision Tree as in Table 1 are used.

improved system is the one that uses Neural Net, which obtains nearly 2% increase in accuracy from the baseline system. Indeed, the Neural Net classifier outperforms the baseline on almost all metrics considered.

Model	Overall	- Negation	- # Tokens	- Bigram	- POS	- NRC
SVM	0.841	0.829	0.844	**0.821**	0.835	0.835
Naïve Bayes	0.835	0.820	0.820	**0.800**	0.849	0.822
Decision Tree	0.844	0.839	0.841	0.837	**0.827**	0.836
Neural Net	0.866	0.862	0.853	0.861	0.872	**0.849**

Table 4: Ablation study of the improved methods. Accuracy is employed for the metric to compare. The evaluation dataset is C-U Yelp review dataset. The numbers in bold represent the greatest drop in accuracy for the current model in case a feature is dropped (i.e., in the row). All implementations are from Python Scikit-Learn package. The same regularization parameters for SVM and Decision Tree as in Table 1 are used.

In Table 4, an ablation study is performed on the Yelp review dataset. Different from the previous scenario, negation does improve the performance here. For the Yelp review dataset, the most influential feature for SVM and Naïve Bayes is the bigram PMI scores. The number of positive and negative tokens is not too helpful for SVM because the influence of repetition of synonyms is small given a long text as in this dataset. On the other hand, the bigram scores can be more accurately estimated, resulting in a much more influential feature. The most influential feature for Decision Tree is the still the POS information, as is the case in the movie review dataset. For NN, the NRC lexicon plays the most important role. The first reason is probably that there is a higher correspondence between the Yelp review dataset and the tweets than the movie reviews. Another reason is that the lexicon information becomes more useful when the length of text is large, thus increasing the hitting probability on the lexicon.

6 Discussion and Conclusions

In this report, a rule-based sentence-level sentiment analysis system was built and analyzed. A baseline system using bag-of-words and four improved systems were proposed. The performance was evaluated and analyzed on the movie review dataset.

One potential improvement is to use word embeddings (such as GloVe) for this task. Currently, only the scores of known words are summed up in a given test sentence, because there is little semantic information for unseen words. Moreover, only surface-level word information (i.e., word forms) is used as input.[2] Suppose that we have a word embedding that is trained on millions of words, there would be not only helpful information for unseen words but also better semantic representation for the existing words.

Another potential improvement is to find better aggregation function to combine word-level scores to sentence-level scores. Currently, sum, max, and min scores, together with the total number of positive and negative words, are used as representing features for the entire sentence, but this is far from complete. It would be very beneficial to train a machine learning algorithm, especially sequential methods such as exponentially weighted average or LSTMs, to learn a function mapping from word-level to sentence-level scores.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mickel Hoang, Oskar Alija Bihorac, and Jacobo Rouces. 2019. Aspect-based sentiment analysis using BERT. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 187–196, Turku, Finland. Linköping University Electronic Press.

Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. 2014. Nrc-canada-2014: Detecting aspects and sentiment in customer reviews. pages 437–442.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 4068–4074. AAAI Press.

Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 321–327, Atlanta, Georgia, USA. Association for Computational Linguistics.

Saif M. Mohammad. 2020. Sentiment analysis: Automatically detecting valence, emotions, and other affectual states from text.

[2] Side remark: using the root of words would indeed worsen the performance in the experiment.

Budi M Mulyo and Dwi H Widyantoro. 2018. Aspect-based sentiment analysis approach with CNN. In *2018 5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, pages 142–147.

Preslav Nakov, Sara Rosenthal, Zornitsa Kozareva, Veselin Stoyanov, Alan Ritter, and Theresa Wilson. 2013. SemEval-2013 task 2: Sentiment analysis in Twitter. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 312–320, Atlanta, Georgia, USA. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278, Barcelona, Spain.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016. Effective LSTMs for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307, Osaka, Japan. The COLING 2016 Organizing Committee.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1422–1432, Lisbon, Portugal. Association for Computational Linguistics.

Sida Wang and Christopher D. Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, page 90–94, USA. Association for Computational Linguistics.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615, Austin, Texas. Association for Computational Linguistics.