

Sentiment Classification and Analysis in Twitter

Justin Cosentino

Department of Computer Science
Swarthmore College
Swarthmore, PA 19081
jcosent1@swarthmore.edu

Emanuel Schorsch

Department of Computer Science
Swarthmore College
Swarthmore, PA 19081
eschors1@swarthmore.edu

Abstract

As the popularity of microblogging drastically increases, it is evident that there is a vast amount of opinionated data and information available to assess the general sentiment of millions of Twitter users in regards to products, events, and people. We consider the problem of classifying the overall sentiment of subjects contained within tweets and the sentiment of a marked instance of a word or phrase within a given tweet as either positive, negative, neutral, or objective.

Using two Twitter corpora provided by SemEval-2013, we implemented and trained a naive Bayes classifier, a decision list classifier, and a subjectivity lexicon classifier. Our data suggests that the accuracy of our models will further increase as the size of our training corpora increases. We also find that both naive Bayes and decision list classifiers trained on a sufficient large n-gram feature set perform far superior to the subjectivity lexicon classifier. We find that the presence of slang and abbreviated words in tweets complicates efforts to build of previous sentiment analysis. Finally, we discuss the issues surrounding the heavy skew of the polarity of tweets towards the most frequent sense.

1 Introduction

To answer the question of how a given population feels about an issue it is usually necessary to conduct some kind of poll. People need to be surveyed and surveyors need to be hired to do the surveying and other laborious tasks. Ideally, we would instead be able to harvest the abundant information on the internet to find our answer.

In recent years we have seen an explosion in social media use. Facebook and Twitter are two prime examples of platforms which millions of different people use on a daily basis. These platforms are perfectly positioned to answer the kinds of questions previously addressed by polls. Since everyone from your teacher to Obama is using these social networking platforms we have access to the opinions of a diverse sample of individuals. We choose to focus on Twitter as it is geared exclusively towards the expression of people's thoughts.

Twitter is a microblogging platform which allows users to broadcast, or tweet, messages (tweets) of up to 140 characters to their followers. Since tweets must be so short they are almost always about a single topic, with usually only a single sentiment expressed. While Twitter does have a significant portion of non-English speaking users, for the purposes of this paper we restrict our attention to tweets in english. Another useful feature of Twitter is that tweets are labeled by hashtags. This makes it simple to search for a specific hashtag such as #Toyota, which will yield all tweets that are self-classified as relating to Toyotas. Given the enormous volume of

tweets per day (340 million¹) it is possible to find a sufficiently large sample of tweets about most topics, even relatively obscure ones.

There remain issues with ensuring that Twitter users represent a sufficiently random population but the issue of automated sentiment extraction looms much larger. In this paper we first discuss how to capture the sentiment of the component phrases of the tweets. We refer to this problem as taskA. This is useful as a building block which can hopefully aid in deciphering the sentiment of a tweet as a whole. Naturally we then consider the tweet as a whole and the overall sentiment of the tweet with respect to a given topic. We refer to this second problem in the paper as taskB. A system which could perform well on this task would be invaluable for anyone studying public opinion. This would be immediately applicable to product managers and campaign managers as well as many researchers.

We found that simple features were sufficient to perform significantly above the baseline. It was interesting to see that having statistics on the prevalence of individual words among different senses was very powerful. We will discuss three different linguistic techniques we used to categorize tweets into their appropriate sense.

2 Related Works

Sentiment analysis is a rapidly expanding field of natural language processing and a vast majority of these studies attempt to tackle a combination of two issues found in sentiment analysis: the identification of sentiment and then the classification of this expressed sentiment. Many experiments attempt to classify the polarity of larger documents, specifically online reviews. For example, Pang and Lee analyze sentiment within online movie reviews (Pang et al., 2002). Using machine learning techniques, they study the effectiveness of a variety of features on sentiment detection. Their studies show that the presence of n-gram models is most useful in determining the polarity of a given review. Additionally, Pang and Lee found that the usage of parts of speech tagging and the presence of adjectives have little effect on improving the accuracy of their models and

decreases this accuracy in some situations. Because of these findings, we have chosen not to experiment with features involving parts of speech. Pang and Lee also chose to experiment with word position. However, because the average tweet contains only eleven words, this is thought to have little effect on determining sentiment and is thus chosen not to be studied in our experiments (O'Connor et al., 2010).

Additionally, there have been many recent studies that attempt to analyze the sentiment contained within tweets and other microblogging message formats. There have been various applications of Twitter sentiment classification, ranging from using microblogging as a form of sharing consumer opinions to linking Twitter sentiment and political public opinion (Jansen et al., 2009; O'Connor et al., 2010). Microblogging features, such as hash-tags and emoticons, have played a key role in assisting in this analysis. Davidov et al. (2010) proposed a supervised sentiment classification framework that utilized these features. Using hand tagged hash-tags and emoticons paired with feature types such as punctuation, patterns, and n-grams, they showed that by using a k-nearest neighbors strategy tweets can be paired with one or more of their labeled hash-tags, giving them the overall sentiment of that tweet. Similarly, Kouloumpis et al. (2011) used microblogging, lexicon, n-gram, and part of speech features to evaluate twitter sentiment. This study confirmed that part of speech features result in a drop in performance in sentiment analysis. Much like our study, these experiments attempt to label tweets as not only containing sentiment, but classifying this polarity as positive, negative, or neutral.

3 Methodology

3.1 Data

A total of three different corpora were used in our experiments. Two of these corpora contained Twitter data that was used in the training and testing of our classifiers. The third data set was comprised of a sentiment lexicon which was used as a form of message classification.

Twitter Corpus

As previously stated, Twitter is a microblogging service that allows users to post short, 140-character

¹<http://blog.twitter.com/2012/03/twitter-turns-six.html>

messages called tweets. The two Twitter corpora used in this study were comprised of such tweets and acquired from the SemEval-2013 Competition site² using the provided script, which downloaded the tweets from the Twitter webpage. Some of these tweets were no longer available due to a user changing their privacy settings, deleting the designated tweet, or deleting their account. These tweets were ignored and not used within our study. Additionally, some tweets contained newline characters, which were removed from the tweet during the download process. Although the data found within these tweets was formatted differently, each corpus contained tweets covering a wide range of topics including entities, products, and events. The tweets found in the corpora were also exclusively written in English. The first corpus contained full tweets and the polarity tag for a marked instance of a word or phrase within each tweet while the second corpus contained full tweets and the polarity tag of a given topic found within the content of each tweet. There were a total of 2,377 tweet segments or phrases within the first corpus and 592 full tweets in the second and the polarity tags for each corpus were limited to 'positive', 'negative', 'neutral' and 'objective'. Although some of the tweets contained within each corpus were identical, the first corpus prompted users to only look at the given segment of the tweet. Each data set was used individually as both training and testing data throughout the study.

Subjectivity Lexicon

The third corpus used in our experiments contained subjectivity data relating to roughly 6,500 words. This lexicon was acquired from Opinion-Finder, a system that performs subjectivity analysis (Wilson et al., 2005). Each word within this corpus has a corresponding polarity, strength of subjectivity, part-of-speech tag, stem value, and word length. For the purposes of this study, the strength of subjectivity, stem value, and word length were ignored. The polarity and part-of-speech tag were then used as means of classifying the sentiment value of tweets.

²Available at <http://www.cs.york.ac.uk/semEval-2013/task2/>

3.2 Features

In order to implement the sentiment classifiers such that they can label the given data, we used a number of features built from the bag-of-words model. This model is implemented as an unordered listing of features and does not take into account word order or word location. Using this model we create feature vectors that contain n-gram features present in a given tweet. The frequency of these features is also represented by the vector. In the implementation of the decision list and naive Bayes classifiers, unigrams, bigrams, and trigrams were used to train and classify data. In the subjectivity lexicon classifier, the individual words making up the tweet and their associated polarity make up the features of the vector.

3.3 Sentiment Classifiers

In order to determine the sentiment of a given tweet, three algorithms were implemented: a naive Bayes classifier, a decision list classifier, and a subjectivity lexicon classifier.

Naive Bayes

Using the Naive Bayes classifier, a given feature vector \vec{f} is assigned the polarity \hat{s} given by:

$$\hat{s} = \arg \max_{s \in S} P(s|\vec{f})$$

. Using Bayes rule we can rewrite this equation:

$$\hat{s} = \arg \max_{s \in S} \frac{P(\vec{f}|s)P(s)}{P(\vec{f})}.$$

Since $P(\vec{f})$ is constant across all polarities, it does not affect the choice of \hat{s} and we can discard it. This leaves us with:

$$\hat{s} = \arg \max_{s \in S} P(\vec{f}|s)P(s).$$

However, it is very unlikely that we will see exactly our given feature vector \vec{f} with n features again. Thus we naively assume that each of the features in the vector are independent of each other. Under this assumption the below equality holds:

$$P(\vec{f}|s) = \prod_{i=1}^n P(f_i|s).$$

The probability of the vector \vec{f} under the assumption of independence is equal to the product of the independent probability of each feature in the vector. Substituting into our previous equation we arrive at our naive Bayes classifier:

$$\hat{s} = \arg \max_{s \in S} \prod_{i=1}^n P(f_i|s)P(s).$$

The probability of f_i which we use is the maximum likelihood estimate. This is computed by the number of times f_i occurs as sense s over the number of times f_i occurs total. This is represented by:

$$P(f_i|s_i) = \frac{\text{count}(f_i, s)}{\text{count}(f_i)}.$$

The probability of a specific sense s is calculated in a similar manner:

$$P(s) = \frac{\text{count}(s)}{\sum_{s_i \in S} \text{count}(s_i)}.$$

However, in our model we do not multiply by $P(s)$. In our model we use the modified formula:

$$\hat{s} = \arg \max_{s \in S} \prod_{i=1}^n P(f_i|s).$$

The feature we relied on most heavily was a unigram model. However, we also implemented bigram and trigram models to see if adding those features to the naive Bayes classifier would improve performance. While higher order n-gram models blatantly violate the assumption of conditional independence the naive Bayes classifier still performs well.

Decision List

A decision list classifier is equivalent to simple case statements or an extended if-else statement. Decision lists generate a set of rules or conditions, for which there is a single classification associated. These rules are generated from tagged feature vectors and then scored and ordered based on their associated scores. Similar to the approach used by Yarowsky, each feature and value pair is treated as a rule (Yarowsky, 1994). In order to generate the best rule for each possible classification, the following equation is used:

$$\left| \log \left(\frac{P(S_i|f_i)}{P(\text{All other senses}|f_i)} \right) \right|.$$

One important difference to note about our system is that our smoothing method is similar but not add-k smoothing. The smoothing method which we used was to add a value alpha to the denominator and numerator of the scoring equation. This stands in contrast to add-k smoothing which is the same but αV is added to the denominator. Once these rules have been generated from the given feature sets and scored, the decision list creates a list of rules similar to those seen in Table 1 such that the rules are ordered highest to lowest in regards to score. The decision list will then use these rules as long as their respective scores remain greater than or equal to one. At this point, the decision list will then proceed to classify words based on the most frequent sense of all tweets in the given training corpus.

Rule	Polarity
love	⇒ Positive
can't wait	⇒ Positive
looking forward	⇒ Positive
confirmed	⇒ Objective
crash	⇒ Negative
...	⇒ ...
Score < 1	⇒ MFS

Table 1: Example rules generated by the decision list after training on the second Twitter corpus.

Subjectivity Lexicon

A subjectivity lexicon was used to classify and determine the polarity of tweets. The subjectivity lexicon was acquired from OpinionFinder, a list containing roughly 6,500 words, their polarity, and the strength of their subjectivity. This lexicon was then used to determine the polarity of each word within the given tweet or tweet segment. The strength of each word, which was labeled as either strong or weak, was ignored for testing.

Two variations of the subjectivity classifier were implemented. In both, individual counts of the number of positive and negative words for each tweet were then kept. If a tweet or tweet segment contained more positive words than negative words, the tweet was then defined as having a positive polarity. If the tweet contained more negative words than positive words, the tweet was determined to have a negative polarity. However in the first classifier,

if a tweet contained neither any positive words nor any negative words or if the count of positive and negative words were equal, the tweet was labeled as objective. Within this subjectivity lexicon classifier, words were not labeled as neutral. If a tweet did not contain any positive words or negative words or contained an equal number of each, then the second subjectivity lexicon classifier would classify the tweet to be the most frequent sense as extracted from the training data. This second classifier will be referred to as the MFS subjectivity lexicon classifier.

Because the first subjectivity classifier requires no labeled tweets to be used as training data, this classifier was tested on all tweets within each corpus. However, in order to compare the results of the subjectivity lexicon classifier to the results of our other classifiers, the lexicon was also used to label only the test data used by all other classifiers. The MFS subjectivity lexicon classifier was trained and tested on the same data as the decision list and naive Bayes classifiers.

Most Frequent Sense

The most frequent sense (MFS) classifier is used as a baseline for comparison in our experiments. For each corpus, the classifier counts the occurrences of each sense. The classifier then labels all test data with the sense with the highest count. This method is also used in the decision list if the list contains no more rules with scores greater than or equal to one and in the MFS subjectivity lexicon classifier.

4 Results

4.1 Tweet Polarity Classification

The classification accuracies for determining the polarity of a given subject found within a tweet are shown in Table 2. These entries represent the average accuracy of each classifier when trained on the given feature using fivefold cross-validation. It is evident that bag-of-unigram features provided the best classification results, with each classifier surpassing the most frequent sense baseline of 52.12%.

Decision List

The decision list classifier did very well when training on unigrams. However, when trained solely on bigrams or solely on trigrams it performed identically to the most frequent sense baseline. When we

combined unigram features with bigram or trigram features, the accuracy of the decision list performed on par with a decision list trained on unigram features. Varying the alpha, the value used in smoothing, had no significant effect on classification accuracy.

Naive Bayes

The naive Bayes classifier was the most consistent classifier and generated the highest average accuracies. When trained on unigram feature sets, this classifier had similar accuracies to the decision list classifier. However, this classifier was able to score much better than both the decision list and the most frequent sense baseline when trained on bag-of-bigram features. This suggests that the naive Bayes classifier is more adept at determining the polarity of tweets when trained on very little data compared to the other classifiers tested.

Subjectivity Lexicon

The subjectivity lexicon had the lowest accuracy of the three classifiers, scoring just above the average most frequent baseline. Although the lexicon was able to correctly determine the polarity of a given word, it was not able to account for sarcasm or negation. This resulted in the incorrect classification of tweets. It is important to note that the most frequent sense subjectivity classifier had a better average accuracy than both the most frequent sense and subjectivity lexicon classifiers, but not as high as either the naive Bayes nor decision list classifiers.

4.2 Contextual Polarity Classification

The classification accuracies for determining the polarity of a given tweet segment are shown in Table 3, but were significantly worse than the accuracy of these classifiers in determining the polarity of a given subject contained within a tweet. Both the decision list and naive Bayes classifiers obtained an accuracy approximately equal to the most frequent sense baseline, while the accuracies of the subjectivity lexicon and MFS subjectivity lexicon classifiers were far below this baseline. It is evident that varying the features each model trains on has little effect on the accuracy. Each entry within Table 3 represents the average accuracy of each classifier when trained on the given feature using fivefold

Features	Naive Bayes	Decision List	Subj. Lexicon	MFS Subj. Lexicon	MFS
Unigrams	59.32%	58.66%	52.28%	55.47%	52.12%
Bigrams	58.13%	52.12%	N/A	N/A	52.12%
Trigrams	52.20%	52.12%	N/A	N/A	52.12%
Bigrams & Unigrams	59.66%	58.49%	N/A	N/A	52.12%
Bi, Tri, & Unigrams	58.48%	58.49%	N/A	N/A	52.12%

Table 2: Average accuracy of classifiers on taskB using fivefold cross-validation on the second Twitter corpus.

Features	Naive Bayes	Decision List	Subj. Lexicon	MFS Subj. Lexicon	MFS
Unigrams	64.84%	64.12%	44.98%	44.98%	64.12%
Bigrams	64.76%	64.12%	N/A	N/A	64.12%
Trigrams	64.38%	64.12%	N/A	N/A	64.12%
Bigrams & Unigrams	65.56%	64.12%	N/A	N/A	64.12%
Bi, Tri, & Unigrams	65.56%	64.12%	N/A	N/A	64.12%

Table 3: Average accuracy of classifiers on taskA using fivefold cross-validation on the first Twitter corpus.

cross-validation.

4.2.1 Decision List

The decision list classifier was only able to match the accuracy of the most frequent sense classifier when determining the polarity of a given subsection of a tweet. Differing the features the classifier was trained on did not result in any differences in this accuracy. Additionally varying alpha, the value used for smoothing, had no effect. This data suggests that training on only small segments of tweets, sometimes only consisting of one word, does not generate rules that will be seen again in other small segments of tweets. Thus, the decision list must tag all tweets with the most frequent sense, regardless of the n-gram features that the classifier is trained on.

4.2.2 Naive Bayes

The naive Bayes classifier was the only classifier able to score above the most frequent sense baseline. However, the accuracy of this classifier was barely above the baseline accuracy of 64.12%. Thus, although this classifier scored slightly higher than the decision list, it is no way an effective means of classifying the polarity of a given segment of a tweet.

4.2.3 Subjectivity Lexicon

Both the subjectivity lexicon and MFS subjectivity lexicon scored far below the baseline of the

most frequent sense classifier. It is evident that these classifiers were falsely identifying sentiment within each tweet segment. Because the most frequent sense for this data set was objective, both classifiers, had they found no sentiment in the tweet or found an equal number of both positive and negative words, would have labeled the polarity of the tweet segment as objective. Thus this low score suggests that the lexicon classifiers were overweighting the sentiment found within these phrases.

4.3 Effects of Limited Training on Accuracy

In addition to analyzing the ability of each classifier to determine the polarity of subjects and segments of tweets, we experimented with limiting the amount of training data available to each classifier. As shown in Figure 1, each classifier was given a percentage of training data from the second corpus and then used to classify the polarity of a designated test data set. For this experiment, twenty-percent of the tweets from the second Twitter corpus were reserved as test data. Each classifier then was trained on another percentage of the corpus, which increased by ten-percent each step. As the data incremented, the most frequent sense and both subjectivity lexicon classifiers remained at a constant accuracy. However, we can see an increase in accuracy in both the naive Bayes and decision list classifiers as the size of the training data increases. This suggests that as each

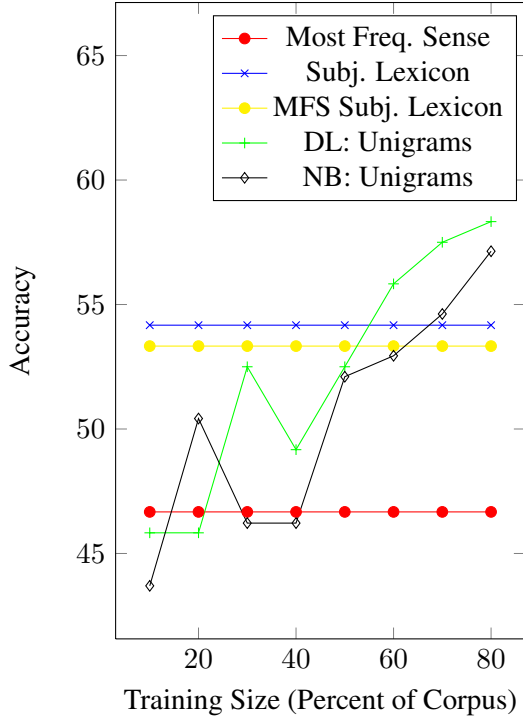


Figure 1: Accuracy of classifiers labeling the sentiment of a given topic contained within a tweet. Each classifier was trained on the indicated percent of the second Twitter corpus and were then tested on the last 20% of the data.

model trains on more data, their accuracy will continue to increase.

5 Analysis

The results we produced using our machine learning algorithms were superior to the baseline of most frequent sense in taskB. However, in taskA the vast majority of our systems only equaled the most frequent sense. One reason for this is that the subset of tweets which we are classifying in taskA are simply too small. This is especially a factor in the decision list. The decision list will be looking for rules with a sufficiently high score. Since there are so few features in each subset it is highly probably that no rule will be triggered. This becomes even more likely when considering the size of our data and hence the smaller number of high quality rules. Since the decision list defaults to the most frequent sense this explains why the decision list is on par with the most frequent sense.

The naive Bayes performs on par with MFS for different reasons though. Since a large majority of the unigrams have been seen before, naive Bayes does have probabilities for most of the features in the test for taskA. This means the naive Bayes is not having the same behavior as the decision list and simply defaulting to the most frequent sense. The problem the naive Bayes system is likely facing is that the training data is almost two thirds objective. The denominator of $p(f_i|s)$ using the maximum likelihood estimate is going to be much larger than the numerator for senses other than objective. This is because the other senses themselves do not appear as often so even if f_i is a great indicator of being "neutral" it will not be reflected in the probability since "neutral" almost never appears.

Despite this shortcoming the naive Bayes algorithm still manages to outperform the most frequent sense baseline by a small margin. As seen in Table 3 using bigrams and unigrams results in an accuracy 1.4% greater than the baseline. While this is not a very large margin (it means only one more correct tag) it is possibly significant as this is an average across the five runs from fivefold cross-validation. This is comforting that the system can indeed label some text with labels other than the most frequent sense despite the large skew of the corpus. The guesses of the naive Bayes system are almost entirely the most frequent sense. There are only a few departures where the system tags tweet subsets as either "positive" or "negative", which accounts for the performance above the most frequent sense baseline. This leaves us with the question of why naive Bayes is consistently outperforming the decision list.

The fundamental feature of decision lists are that they have a list of rules and simply go through them and execute any rules that apply. Since rules are based on features which almost always include unigrams it is likely that at least one of the words in the tweet will trigger a rule. The rules in the decision list for taskB are almost entirely objective rules, especially for the highest scoring rules. This is due to the uneven distribution of classes previously discussed. What this means is that since the algorithm descends down the list of rules it is likely to see an applicable rule at the very top, which will then likely be "objective". The decision list does not take into account if the next five rules in the decision list were all for

”positive” and they would have all been triggered. This is seen in the data by the fact that not a single time in the five runs of the fivefold cross-validation did the decision list classify any tweet subset as being anything other than objective.

It seems decision lists work well when rules exist with very high separating power, meaning if a rule is triggered the object almost certainly belongs in the specified class. In Twitter though most words have multiple meanings not all of which belong to the same polarity class. More importantly though is the general concept of negation. Since any word can be negated, if not directly then in a rhetorical question, no word in the English language will belong solely to one class i.e. thwarted expectations³ (DO WE WANT TO BASICALLY COPY THE ANALYSIS OUT OF THE OTHER PAPER). The features which are most indicative of a class, such as extremely positive words like ”love” and ”awesome”, are precisely those features which are most likely to be negated to express the opposite sentiment. The decision list is then expressing 100% confidence when it encounters these rules despite them having a not insignificant margin of error.

On the other hand, naive Bayes allows for much more flexibility. Even if a given word has a strong correlation with being ”positive” the tweet can still be labeled a different sense if a sufficient number of other words are associated with a different polarity. These words do not have to be in close proximity to each other which allows the naive Bayes to take into account context in a natural way. In sharp contrast the decision list only accounts for context if the included features have some notion of context included in them, which unigrams certainly do not.

Both naive Bayes and the decision list performed very significantly better than the subjectivity lexicons. This makes perfect sense as the subjectivity lexicons were using a sentiment dictionary derived from newspaper texts. The writing in newspapers differs markedly from tweets. The most notable difference is the absence of slang and emoticons. However, naive Bayes and the decision list will notice apparent sentiments of slang words and emoticons as the probabilities are being generated from real tweets and so reflect the way the language is actu-

ally used.

We attempted to rectify the lack of context in decision lists by incorporating bigrams and trigrams. For bigrams or trigrams alone the training data was too small so the decision list resorted to labeling with the most frequent sense every time. However, what was surprising is that combining unigrams and bigrams did not improve the decision list performance and even made it slightly worse. It seems then that bigrams do not capture context well. By the fact that the performance did change slightly with the addition of bigrams we know that some bigrams ended up in the decision list and were triggered. One would think that bigrams would be more accurate than unigrams. This failure is surprising until we remember that in the bigrams defense they likely all have very low counts. Since our training set was not very large it is unlikely that any bigram was seen too many times meaning any given bigram’s distribution among the senses could easily be a fluke. Given each bigram’s small number of occurrences it is probable that a few of the bigrams fell all in one sense. These bigrams would then be added as incredibly accurate rules despite their suspect nature.

Since naive Bayes does not have the same problem with unjustified early commitment the suspect nature of bigrams is not a large problem. When bigrams are combined with unigrams in naive Bayes in both tasks naive Bayes outperforms using unigrams alone. This reflects the fact that in aggregate the bigrams are more accurate than the unigrams. Since the naive Bayes is multiplying so many terms it is not unduly affected by a few incorrect terms. We note though, that when using both bigrams and unigrams the naive Bayes system consistently guesses more words as ”objective”, the most frequent sense. Since we are already performing above the most frequent sense baseline this means that the bigrams are correcting some over-eager unigram tagging.

Recall the unique smoothing algorithm we used. When we used add-k smoothing our performances significantly dropped across the board. The reason is that our decision list defaults to most frequent sense once the scores fall below a cutoff of 1. As V gets larger the scores of the rules become lower. This means the decision list resorts to the default more readily, decreasing accuracy.

An additional quirk of our system which we found

³see Pang paper pg 7 for further explanation

is that the naive Bayes model performs much better when using our modified formula. Originally we were using the real naive Bayes formula and multiplying by $P(s)$. We suspected the performance might have improved when not multiplying by $P(s)$ since then senses other than the most frequent sense would have a chance to be chosen. However, on examination it actually appears that way random senses are chosen when $P(s)$ is included. Our system then becomes way more selective about deviating from the most frequent sense and therefore outperforms our baseline.

We are clearly relying on knowledge of the most frequent sense from our training data. However, we believe this is not a problem as the most frequent sense is so much more common than the rest as to suggest that it is a feature of Twitter and not our corpus. Since our systems are fine-tuned specifically for Twitter this is no more of a problem than taking advantage of our knowledge that the tweets we are analysing are in english.

6 Conclusion

It is clear that in both taskA and taskB the naive Bayes and the decision list are very conservative systems which guess the most frequent sense unless they are fairly confident that a different sense is correct. Over 80% of the guesses are in the most frequent sense. This produces good results as the systems perform at or above the baselines. This is especially noticeable in taskB where both systems significantly outperform the most frequent sense baseline.

Our subjectivity lexicon performs quite poorly. This is due to the lexicon being very inapplicable due to its being extracted from newspapers. Despite the limited size of our training data both naive Bayes and the decision list were able to learn appropriate rules. The results of training those two systems on progressively more data showed that their performance will scale if we get more data to train on.

6.1 Future Work

One shortcoming to our sentiment lexicon is that the sentiment words weren't derived from tweets. This is a large problem since word usage changes dramatically from written text to internet micro-blogging.

In order to account for slang, emoticons, and other idiosyncracies of Twitter it would be interesting to implement a bootstrapping algorithm (Riloff et al., 2003). This would allow us to extract sentiment words directly from training data increasing the accuracy of the sentiment lexicon.

Another difficulty we faced is the uneven distribution of the class. A potential solution to this is to implement the modified naive bayes discussed by (Frank and Bouckaert, 2006). This is a large issue as even in taskB we still have one class accounting for over half of all occurrences. The authors show a promising solution involving a normalization step during the smoothing process which reduces the impact of the class skew.

Finally, addressing the issue of negation would significantly improve our performance. Implementing some kind of rule based system could possibly correct for our system thinking "not great" is positive. Many of our errors were due to negations such as the one previously mentioned so even a naive fix for this could lead to improvements.

References

- D. Davidov, O. Tsur, and A. Rappoport. 2010. Enhanced sentiment learning using twitter hashtags and smileys. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 241–249. Association for Computational Linguistics.
- E. Frank and R. Bouckaert. 2006. Naive bayes for text classification with unbalanced classes. In *Proc 10th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 503–510.
- B.J. Jansen, M. Zhang, K. Sobel, and A. Chowdury. 2009. Twitter power: Tweets as electronic word of mouth. *Journal of the American society for information science and technology*, 60(11):2169–2188.
- E. Kouloumpis, T. Wilson, and J. Moore. 2011. Twitter sentiment analysis: The good the bad and the omg. In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*.
- B. O'Connor, R. Balasubramanian, B.R. Routledge, and N.A. Smith. 2010. From tweets to polls: Linking text sentiment to public opinion time series. In *Proceedings of the International AAAI Conference on Weblogs and Social Media*, pages 122–129.
- B. Pang, L. Lee, and S. Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning

- techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics.
- E. Riloff, T. Wiebe, and T. Wilson. 2003. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the Seventh CoNLL conference held at HLT-NAACL 2003*, pages 25–32.
- T. Wilson, P. Hoffmann, S. Somasundaran, J. Kessler, J. Wiebe, Y. Choi, C. Cardie, E. Riloff, and S. Patwardhan. 2005. Opinionfinder: A system for subjectivity analysis. In *Proceedings of HLT/EMNLP on Interactive Demonstrations*, pages 34–35. Association for Computational Linguistics.
- D. Yarowsky. 1994. Decision lists for lexical ambiguity resolution: Application to accent restoration in spanish and french. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 88–95. Association for Computational Linguistics.