Sentiment Analysis of Twitter Data

Apoorv Agarwal Boyi Xie Ilia Vovsha Owen Rambow Rebecca Passonneau
Department of Computer Science
Columbia University
New York, NY 10027 USA
{apoorv@cs, xie@cs, iv2121@, rambow@ccls, becky@cs}.columbia.edu

Abstract

We examine sentiment analysis on Twitter data. The contributions of this paper are: (1) We introduce POS-specific prior polarity features. (2) We explore the use of a tree kernel to obviate the need for tedious feature engineering. The new features (in conjunction with previously proposed features) and the tree kernel perform approximately at the same level, both outperforming the state-of-the-art baseline.

1 Introduction

Microblogging websites have evolved to become a source of varied kind of information. This is due to nature of microblogs on which people post real time messages about their opinions on a variety of topics, discuss current issues, complain, and express positive sentiment for products they use in daily life. In fact, companies manufacturing such products have started to poll these microblogs to get a sense of general sentiment for their product. Many times these companies study user reactions and reply to users on microblogs. One challenge is to build technology to detect and summarize an overall sentiment.

In this paper, we look at one such popular microblog called Twitter and build models for classifying "tweets" into positive, negative and neutral sentiment. We build models for two classification tasks: a binary task of classifying sentiment into positive and negative classes and a 3-way task of classifying sentiment into positive, negative and neutral classes. We experiment with three types of models: unigram model, a feature based model and a tree kernel based model. For the feature based model we use some of the features proposed in past literature and propose new features. For the tree kernel based model we design a new tree representation for tweets. We use a unigram model, previously shown to work well for sentiment analysis for Twitter data, as our baseline. Our experiments show that a unigram model is indeed a hard baseline achieving over 20% over the chance baseline for both classification tasks. Our feature based model that uses only 100 features achieves similar accuracy as the unigram model that uses over 10,000 features. Our tree kernel based model outperforms both these models by a significant margin. We also experiment with a combination of models: combining unigrams with our features and combining our features with the tree kernel. Both these combinations outperform the unigram baseline by over 4% for both classification tasks. In this paper, we present extensive feature analysis of the 100 features we propose. Our experiments show that features that have to do with Twitter-specific features (emoticons, hashtags etc.) add value to the classifier but only marginally. Features that combine prior polarity of words with their parts-of-speech tags are most important for both the classification tasks. Thus, we see that standard natural language processing tools are useful even in a genre which is quite different from the genre on which they were trained (newswire). Furthermore, we also show that the tree kernel model performs roughly as well as the best feature based models, even though it does not require detailed feature engineering.

We use manually annotated Twitter data for our

Proceedings of the Workshop on Language in Social Media (LSM 2011), pages 30–38,
Portland, Oregon, 23 June 2011.

experiments. One advantage of this data, over previously used data-sets, is that the tweets are collected in a streaming fashion and therefore represent a true sample of actual tweets in terms of language use and content. Our new data set is available to other researchers. In this paper we also introduce two resources which are available (contact the first author): 1) a hand annotated dictionary for emoticons that maps emoticons to their polarity and 2) an acronym dictionary collected from the web with English translations of over 5000 frequently used acronyms.

The rest of the paper is organized as follows. In section 2, we discuss classification tasks like sentiment analysis on micro-blog data. In section 3, we give details about the data. In section 4 we discuss our pre-processing technique and additional resources. In section 5 we present our prior polarity scoring scheme. In section 6 we present the design of our tree kernel. In section 7 we give details of our feature based approach. In section 8 we present our experiments and discuss the results. We conclude and give future directions of research in section 9.

2 Literature Survey

Sentiment analysis has been handled as a Natural Language Processing task at many levels of granularity. Starting from being a document level classification task (Turney, 2002; Pang and Lee, 2004), it has been handled at the sentence level (Hu and Liu, 2004; Kim and Hovy, 2004) and more recently at the phrase level (Wilson et al., 2005; Agarwal et al., 2009).

Microblog data like Twitter, on which users post real time reactions to and opinions about "everything", poses newer and different challenges. Some of the early and recent results on sentiment analysis of Twitter data are by Go et al. (2009), (Bermingham and Smeaton, 2010) and Pak and Paroubek (2010). Go et al. (2009) use distant learning to acquire sentiment data. They use tweets ending in positive emoticons like ":)" ":-)" as positive and negative emoticons like ":(" ":-(" as negative. They build models using Naive Bayes, MaxEnt and Support Vector Machines (SVM), and they report SVM outperforms other classifiers. In terms of feature space, they try a Unigram, Bigram model in conjunction with parts-of-speech (POS) features. They note that the unigram model outperforms all other models. Specifically, bigrams and POS features do not help. Pak and Paroubek (2010) collect data following a similar distant learning paradigm. They perform a different classification task though: subjective versus objective. For subjective data they collect the tweets ending with emoticons in the same manner as Go et al. (2009). For objective data they crawl twitter accounts of popular newspapers like "New York Times", "Washington Posts" etc. They report that POS and bigrams both help (contrary to results presented by Go et al. (2009)). Both these approaches, however, are primarily based on ngram models. Moreover, the data they use for training and testing is collected by search queries and is therefore biased. In contrast, we present features that achieve a significant gain over a unigram baseline. In addition we explore a different method of data representation and report significant improvement over the unigram models. Another contribution of this paper is that we report results on manually annotated data that does not suffer from any known biases. Our data is a random sample of streaming tweets unlike data collected by using specific queries. The size of our hand-labeled data allows us to perform cross-validation experiments and check for the variance in performance of the classifier across folds.

Another significant effort for sentiment classification on Twitter data is by Barbosa and Feng (2010). They use polarity predictions from three websites as noisy labels to train a model and use 1000 manually labeled tweets for tuning and another 1000 manually labeled tweets for testing. They however do not mention how they collect their test data. They propose the use of syntax features of tweets like retweet, hashtags, link, punctuation and exclamation marks in conjunction with features like prior polarity of words and POS of words. We extend their approach by using real valued prior polarity, and by combining prior polarity with POS. Our results show that the features that enhance the performance of our classifiers the most are features that combine prior polarity of words with their parts of speech. The tweet syntax features help but only marginally.

Gamon (2004) perform sentiment analysis on feeadback data from Global Support Services survey. One aim of their paper is to analyze the role

of linguistic features like POS tags. They perform extensive feature analysis and feature selection and demonstrate that abstract linguistic analysis features contributes to the classifier accuracy. In this paper we perform extensive feature analysis and show that the use of only 100 abstract linguistic features performs as well as a hard unigram baseline.

3 Data Description

Twitter is a social networking and microblogging service that allows users to post real time messages, called tweets. Tweets are short messages, restricted to 140 characters in length. Due to the nature of this microblogging service (quick and short messages), people use acronyms, make spelling mistakes, use emoticons and other characters that express special meanings. Following is a brief terminology associated with tweets. Emoticons: These are facial expressions pictorially represented using punctuation and letters; they express the user's mood. Target: Users of Twitter use the "@" symbol to refer to other users on the microblog. Referring to other users in this manner automatically alerts them. Hashtags: Users usually use hashtags to mark topics. This is primarily done to increase the visibility of their tweets.

We acquire 11,875 manually annotated Twitter data (tweets) from a commercial source. They have made part of their data publicly available. For information on how to obtain the data, see Acknowledgments section at the end of the paper. They collected the data by archiving the real-time stream. No language, location or any other kind of restriction was made during the streaming process. In fact, their collection consists of tweets in foreign languages. They use Google translate to convert it into English before the annotation process. Each tweet is labeled by a human annotator as *positive*, *negative*, *neutral* or *junk*. The "junk" label means that the tweet cannot be understood by a human annotator. A manual analysis of a random sample of tweets labeled as "junk" suggested that many of these tweets were those that were not translated well using Google translate. We eliminate the tweets with *junk* label for experiments. This leaves us with an unbalanced sample of 8,753 tweets. We use stratified sampling to get a balanced data-set of 5127 tweets (1709 tweets each from classes positive, negative and neutral).

4 Resources and Pre-processing of data

In this paper we introduce two new resources for pre-processing twitter data: 1) an emoticon dictionary and 2) an acronym dictionary. We prepare the emoticon dictionary by labeling 170 emoticons listed on Wikipedia[1] with their emotional state. For example, ":)" is labeled as positive whereas ":=(" is labeled as negative. We assign each emoticon a label from the following set of labels: Extremely-positive, Extremely-negative, Positive, Negative, and Neutral. We compile an acronym dictionary from an online resource.[2] The dictionary has translations for 5,184 acronyms. For example, *lol* is translated to *laughing out loud*.

We pre-process all the tweets as follows: a) replace all the emoticons with a their sentiment polarity by looking up the emoticon dictionary, b) replace all URLs with a tag $||U||$, c) replace targets (e.g. "@John") with tag $||T||$, d) replace all negations (e.g. *not*, *no*, *never*, *n't*, *cannot*) by tag "NOT", and e) replace a sequence of repeated characters by three characters, for example, convert *cooooooooool* to *coool*. We do not replace the sequence by only two characters since we want to differentiate between the regular usage and emphasized usage of the word.

Acronym	English expansion
gr8, gr8t	great
lol	laughing out loud
rotf	rolling on the floor
bff	best friend forever

Table 1: Example acrynom and their expansion in the acronym dictionary.

We present some preliminary statistics about the data in Table 3. We use the Stanford tokenizer (Klein and Manning, 2003) to tokenize the tweets. We use a stop word dictionary[3] to identify stop words. All the other words which are found in WordNet (Fellbaum, 1998) are counted as English words. We use

[1] http://en.wikipedia.org/wiki/List_of_emoticons
[2] http://www.noslang.com/
[3] http://www.webconfs.com/stop-words.php

Emoticon	Polarity
:-) :) :o) :] :3 :c)	Positive
:D C:	Extremely-Positive
:-(:(:c :[	Negative
D8 D; D= DX v.v	Extremely-Negative
: |	Neutral

Table 2: Part of the dictionary of emoticons

the standard tagset defined by the Penn Treebank for identifying punctuation. We record the occurrence of three standard twitter tags: emoticons, URLs and targets. The remaining tokens are either non English words (like *coool*, *zzz* etc.) or other symbols.

Number of tokens	79,152
Number of stop words	30,371
Number of English words	23,837
Number of punctuation marks	9,356
Number of capitalized words	4,851
Number of twitter tags	3,371
Number of exclamation marks	2,228
Number of negations	942
Number of other tokens	9047

Table 3: Statistics about the data used for our experiments.

In Table 3 we see that 38.3% of the tokens are stop words, 30.1% of the tokens are found in WordNet and 1.2% tokens are negation words. 11.8% of all the tokens are punctuation marks excluding exclamation marks which make up for 2.8% of all tokens. In total, 84.1% of all tokens are tokens that we expect to see in a typical English language text. There are 4.2% tags that are specific to Twitter which include emoticons, target, hastags and "RT" (retweet). The remaining 11.7% tokens are either words that cannot be found in WordNet (like *Zzzzz*, *kewl*) or special symbols which do not fall in the category of Twitter tags.

5 Prior polarity scoring

A number of our features are based on prior polarity of words. For obtaining the prior polarity of words, we take motivation from work by Agarwal et al. (2009). We use Dictionary of Affect in Language (DAL) (Whissel, 1989) and extend it using WordNet. This dictionary of about 8000 English language words assigns every word a pleasantness score ($\in \mathbb{R}$) between 1 (Negative) - 3 (Positive). We first normalize the scores by diving each score my the scale (which is equal to 3). We consider words with polarity less than 0.5 as negative, higher than 0.8 as positive and the rest as neutral. If a word is not directly found in the dictionary, we retrieve all synonyms from Wordnet. We then look for each of the synonyms in DAL. If any synonym is found in DAL, we assign the original word the same pleasantness score as its synonym. If none of the synonyms is present in DAL, the word is not associated with any prior polarity. For the given data we directly found prior polarity of 81.1% of the words. We find polarity of other 7.8% of the words by using WordNet. So we find prior polarity of about 88.9% of English language words.

6 Design of Tree Kernel

We design a tree representation of tweets to combine many categories of features in one succinct convenient representation. For calculating the similarity between two trees we use a Partial Tree (PT) kernel first proposed by Moschitti (2006). A PT kernel calculates the similarity between two trees by comparing all possible sub-trees. This tree kernel is an instance of a general class of convolution kernels. Convolution Kernels, first introduced by Haussler (1999), can be used to compare abstract objects, like strings, instead of feature vectors. This is because these kernels involve a recursive calculation over the "parts" of abstract object. This calculation is made computationally efficient by using Dynamic Programming techniques. By considering all possible combinations of fragments, tree kernels capture any possible correlation between features and categories of features.

Figure 1 shows an example of the tree structure we design. This tree is for a synthesized tweet: *@Fernando this isn't a great day for playing the HARP! :)*. We use the following procedure to convert a tweet into a tree representation: Initialize the main tree to be "ROOT". Then tokenize each tweet and for each token: a) if the token is a target, emoticon, exclamation mark, other punctuation mark, or a negation word, add a leaf node to the "ROOT" with

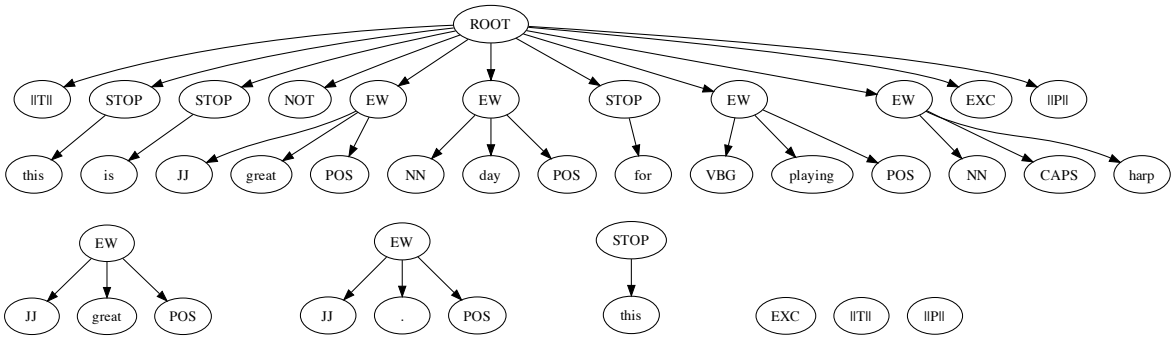

Figure 1: Tree kernel for a synthesized tweet: "@Fernando this isn't a great day for playing the HARP! :)"

the corresponding tag. For example, in the tree in Figure 1 we add tag $||T||$ (target) for "@Fernando", add tag "NOT" for the token "n't", add tag "EXC" for the exclamation mark at the end of the sentence and add $||P||$ for the emoticon representing positive mood. b) if the token is a stop word, we simply add the subtree " (STOP ('stop-word'))" to "ROOT". For instance, we add a subtree corresponding to each of the stop words: *this*, *is*, and *for*. c) if the token is an English language word, we map the word to its part-of-speech tag, calculate the prior polarity of the word using the procedure described in section 5 and add the subtree (EW ('POS' 'word' 'prior polarity')) to the "ROOT". For example, we add the subtree (EW (JJ great POS)) for the word *great*. "EW" refers to English word. d) For any other token <token> we add subtree "(NE (<token>))" to the "ROOT". "NE" refers to non-English.

The PT tree kernel creates all possible subtrees and compares them to each other. These subtrees include subtrees in which non-adjacent branches become adjacent by excising other branches, though order is preserved. In Figure 1, we show some of the tree fragments that the PT kernel will attempt to compare with tree fragments from other trees. For example, given the tree (EW (JJ) (great) (POS)), the PT kernel will use (EW (JJ) (great) (POS)), (EW (great) (POS)), (EW (JJ) (POS)), (EW (JJ) (great)), (EW (JJ)), (EW (great)), (EW (POS)), (EW), (JJ), (great), and (POS). This means that the PT tree kernel attempts to use full information, and also abstracts away from specific information (such as the lexical item). In this manner, it is not necessary to create by hand features at all levels of abstraction.

7 Our features

We propose a set of features listed in Table 4 for our experiments. These are a total of 50 type of features. We calculate these features for the whole tweet and for the last one-third of the tweet. In total we get 100 additional features. We refer to these features as Senti-features throughout the paper.

Our features can be divided into three broad categories: ones that are primarily counts of various features and therefore the value of the feature is a natural number $\in \mathbb{N}$. Second, features whose value is a real number $\in \mathbb{R}$. These are primarily features that capture the score retrieved from DAL. Thirdly, features whose values are boolean $\in \mathbb{B}$. These are bag of words, presence of exclamation marks and capitalized text. Each of these broad categories is divided into two subcategories: Polar features and Non-polar features. We refer to a feature as polar if we calculate its prior polarity either by looking it up in DAL (extended through WordNet) or in the emoticon dictionary. All other features which are not associated with any prior polarity fall in the Non-polar category. Each of Polar and Non-polar features is further subdivided into two categories: POS and Other. POS refers to features that capture statistics about parts-of-speech of words and Other refers to all other types of features.

In reference to Table 4, row f_1 belongs to the category Polar POS and refers to the count of number of positive and negative parts-of-speech (POS) in a tweet, rows f_2, f_3, f_4 belongs to the category Po-

lar Other and refers to count of number of negation words, count of words that have positive and negative prior polarity, count of emoticons per polarity type, count of hashtags, capitalized words and words with exclamation marks associated with words that have prior polarity, row f_5 belongs to the category Non-Polar POS and refers to counts of different parts-of-speech tags, rows f_6, f_7 belong to the category Non-Polar Other and refer to count of number of slangs, latin alphabets, and other words without polarity. It also relates to special terms such as the number of hashtags, URLs, targets and newlines. Row f_8 belongs to the category Polar POS and captures the summation of prior polarity scores of words with POS of JJ, RB, VB and NN. Similarly, row f_9 belongs to the category Polar Other and calculates the summation of prior polarity scores of all words, row f_{10} refers to the category Non-Polar Other and calculates the percentage of tweet that is capitalized.

Finally, row f_{11} belongs to the category Non-Polar Other and refers to presence of exclamation and presence of capitalized words as features.

8 Experiments and Results

In this section, we present experiments and results for two classification tasks: 1) Positive versus Negative and 2) Positive versus Negative versus Neutral. For each of the classification tasks we present three models, as well as results for two combinations of these models:

1. Unigram model (our baseline)
2. Tree kernel model
3. 100 Senti-features model
4. Kernel plus Senti-features
5. Unigram plus Senti-features

For the unigram plus Senti-features model, we present feature analysis to gain insight about what kinds of features are adding most value to the model. We also present learning curves for each of the models and compare learning abilities of models when provided limited data.

Experimental-Set-up: For all our experiments we use Support Vector Machines (SVM) and report averaged 5-fold cross-validation test results. We tune the C parameter for SVM using an embedded 5-fold cross-validation on the training data of each fold, i.e. for each fold, we first run 5-fold cross-validation only on the training data of that fold for different values of C. We pick the setting that yields the best cross-validation error and use that C for determining test error for that fold. As usual, the reported accuracies is the average over the five folds.

8.1 Positive versus Negative

This is a binary classification task with two classes of sentiment polarity: positive and negative. We use a balanced data-set of 1709 instances for each class and therefore the chance baseline is 50%.

8.1.1 Comparison of models

We use a unigram model as our baseline. Researchers report state-of-the-art performance for sentiment analysis on Twitter data using a unigram model (Go et al., 2009; Pak and Paroubek, 2010). Table 5 compares the performance of three models: unigram model, feature based model using only 100 Senti-features, and the tree kernel model. We report mean and standard deviation of 5-fold test accuracy. We observe that the tree kernels outperform the unigram and the Senti-features by 2.58% and 2.66% respectively. The 100 Senti-features described in Table 4 performs as well as the unigram model that uses about 10,000 features. We also experiment with combination of models. Combining unigrams with Senti-features outperforms the combination of kernels with Senti-features by 0.78%. This is our best performing system for the positive versus negative task, gaining about 4.04% absolute gain over a hard unigram baseline.

8.1.2 Feature Analysis

Table 6 presents classifier accuracy and F1-measure when features are added incrementally. We start with our baseline unigram model and subsequently add various sets of features. First, we add all non-polar features (rows $f_5, f_6, f_7, f_{10}, f_{11}$ in Table 4) and observe no improvement in the performance. Next, we add all part-of-speech based features (rows f_1, f_8) and observe a gain of 3.49% over the unigram baseline. We see an additional increase in accuracy by 0.55% when we add other prior polarity features (rows f_2, f_3, f_4, f_9 in Table 4). From

$\mathbb{N}$	Polar	POS	# of (+/-) POS (JJ, RB, VB, NN)	f_1
		Other	# of negation words, positive words, negative words	f_2
			# of extremely-pos., extremely-neg., positive, negative emoticons	f_3
			# of (+/-) hashtags, capitalized words, exclamation words	f_4
	Non-Polar	POS	# of JJ, RB, VB, NN	f_5
		Other	# of slangs, latin alphabets, dictionary words, words	f_6
			# of hashtags, URLs, targets, newlines	f_7
$\mathbb{R}$	Polar	POS	For POS JJ, RB, VB, NN, $\sum$ prior pol. scores of words of that POS	f_8
		Other	$\sum$ prior polarity scores of all words	f_9
	Non-Polar	Other	percentage of capitalized text	f_{10}
$\mathbb{B}$	Non-Polar	Other	exclamation, capitalized text	f_{11}

Table 4: $\mathbb{N}$ refers to set of features whose value is a positive integer. They are primarily count features; for example, count of number of positive adverbs, negative verbs etc. $\mathbb{R}$ refers to features whose value is a real number; for example, sum of the prior polarity scores of words with part-of-speech of adjective/adverb/verb/noun, and sum of prior polarity scores of all words. $\mathbb{B}$ refers to the set of features that have a boolean value; for example, presence of exclamation marks, presence of capitalized text.

Model	Avg. Acc (%)	Std. Dev. (%)
Unigram	71.35	1.95
Senti-features	71.27	0.65
Kernel	**73.93**	1.50
Unigram + Senti-features	**75.39**	1.29
Kernel + Senti-features	74.61	1.43

Table 5: Average and standard deviation for test accuracy for the 2-way classification task using different models: Unigram (baseline), tree kernel, Senti-features, unigram plus Senti-features, and tree kernel plus senti-features.

these experiments we conclude that the most important features in Senti-features are those that involve prior polarity of parts-of-speech. All other features play a marginal role in achieving the best performing system. In fact, we experimented by using unigrams with only prior polarity POS features and achieved a performance of 75.1%, which is only slightly lower than using all Senti-features.

In terms of unigram features, we use Information Gain as the attribute evaluation metric to do feature selection. In Table 7 we present a list of unigrams that consistently appear as top 15 unigram features across all folds. Words having positive or negative prior polarity top the list. Emoticons also appear as important unigrams. Surprisingly though, the word *for* appeared as a top feature. A preliminary analysis revealed that the word *for* appears as frequently in positive tweets as it does in negative tweets. However, tweets containing phrases like *for you* and *for me* tend to be positive even in the absence of any other explicit prior polarity words. Owing to previous research, the URL appearing as a top feature is less surprising because Go et al. (2009) report that tweets containing URLs tend to be positive.

Features	Acc.	F1 Measure	
		Pos	Neg
Unigram baseline	71.35	71.13	71.50
+ $f_5, f_6, f_7, f_{10}, f_{11}$	70.1	69.66	70.46
+ f_1, f_8	74.84	74.4	75.2
+ f_2, f_3, f_4, f_9	**75.39**	74.81	75.86

Table 6: Accuracy and F1-measure for 2-way classification task using Unigrams and Senti-features. All f_i refer to Table 4 and are cumulative.

Positive words	love, great, good, thanks
Negative words	hate, shit, hell, tired
Emoticons	$\|\|P\|\|$ (positive emoticon), $\|\|N\|\|$ (negative emoticon)
Other	for, $\|\|U\|\|$ (URL)

Table 7: List of top unigram features for 2-way task.

8.1.3 Learning curve

The learning curve for the 2-way classification task is in Figure 2. The curve shows that when lim-

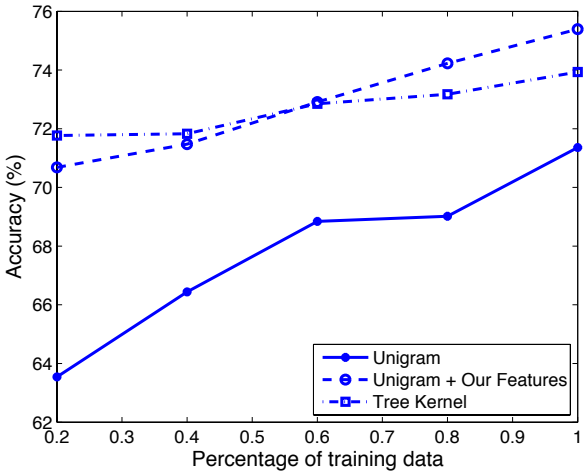

Figure 2: Learning curve for two-way classification task.

ited data is used the advantages in the performance of our best performing systems is even more pronounced. This implies that with limited amount of training data, simply using unigrams has a critical disadvantage, while both tree kernel and unigram model with our features exhibit promising performance.

8.2 Positive versus Negative versus Neutral

This is a 3-way classification task with classes of sentiment polarity: positive, negative and neutral. We use a balanced data-set of 1709 instances for each class and therefore the chance baseline is 33.33%.

8.2.1 Comparison of models

For this task the unigram model achieves a gain of 23.25% over chance baseline. Table 8 compares the performance of our three models. We report mean and standard deviation of 5-fold test accuracy. We observe that the tree kernels outperform the unigram and the Senti-features model by 4.02% and 4.29% absolute, respectively. We note that this difference is much more pronounced comparing to the two way classification task. Once again, our 100 Senti-features perform almost as well as the unigram baseline which has about 13,000 features. We also experiment with the combination of models. For this classification task the combination of tree kernel with Senti-features outperforms the combination of unigrams with Senti-features by a small margin.

Model	Avg. Acc (%)	Std. Dev. (%)
Unigram	56.58	1.52
Senti-features	56.31	0.69
Kernel	**60.60**	1.00
Unigram + Senti-features	60.50	2.27
Kernel + Senti-features	**60.83**	1.09

Table 8: Average and standard deviation for test accuracy for the 3-way classification task using different models: Unigram (baseline), tree kernel, Senti-features, unigram plus Senti-features, and Senti-features plus tree kernels.

This is our best performing system for the 3-way classification task, gaining 4.25% over the unigram baseline.

The learning curve for the 3-way classification task is similar to the curve of the 2-way classification task, and we omit it.

8.2.2 Feature Analysis

Table 9 presents classifier accuracy and F1-measure when features are added incrementally. We start with our baseline unigram model and subsequently add various sets of features. First, we add all non-polar features (rows f_5, f_6, f_7, f_{10} in Table 4) and observe an small improvement in the performance. Next, we add all part-of-speech based features and observe a gain of 3.28% over the unigram baseline. We see an additional increase in accuracy by 0.64% when we add other prior polarity features (rows f_2, f_3, f_4, f_9 in Table 4). These results are in line with our observations for the 2-way classification task. Once again, the main contribution comes from features that involve prior polarity of parts-of-speech.

Features	Acc.	F1 Measure		
		Pos	Neu	Neg
Unigram baseline	56.58	56.86	56.58	56.20
+ $f_5, f_6, f_7, f_{10}, f_{11}$	56.91	55.12	59.84	55
+ f_1, f_8	59.86	58.42	61.04	59.82
+ f_2, f_3, f_4, f_9	**60.50**	59.41	60.15	61.86

Table 9: Accuracy and F1-measure for 3-way classification task using unigrams and Senti-features.

The top ranked unigram features for the 3-way

classification task are mostly similar to that of the 2-way classification task, except several terms with neutral polarity appear to be discriminative features, such as *to*, *have*, and *so*.

9 Conclusion

We presented results for sentiment analysis on Twitter. We use previously proposed state-of-the-art unigram model as our baseline and report an overall gain of over 4% for two classification tasks: a binary, positive versus negative and a 3-way positive versus negative versus neutral. We presented a comprehensive set of experiments for both these tasks on manually annotated data that is a random sample of stream of tweets. We investigated two kinds of models: tree kernel and feature based models and demonstrate that both these models outperform the unigram baseline. For our feature-based approach, we do feature analysis which reveals that the most important features are those that combine the prior polarity of words and their parts-of-speech tags. We tentatively conclude that sentiment analysis for Twitter data is not that different from sentiment analysis for other genres.

In future work, we will explore even richer linguistic analysis, for example, parsing, semantic analysis and topic modeling.

10 Acknowledgments

Agarwal and Rambow are funded by NSF grant IIS-0713548. Vovsha is funded by NSF grant IIS-0916200. We would like to thank NextGen Invent (NGI) Corporation for providing us with the Twitter data. Please contact Deepak Mittal (deepak.mittal@ngicorportion.com) about obtaining the data.

References

Apoorv Agarwal, Fadi Biadsy, and Kathleen Mckeown. 2009. Contextual phrase-level polarity analysis using lexical affect scoring and syntactic n-grams. *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 24–32, March.

Luciano Barbosa and Junlan Feng. 2010. Robust sentiment detection on twitter from biased and noisy data. *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 36–44.

Adam Bermingham and Alan Smeaton. 2010. Classifying sentiment in microblogs: is brevity an advantage is brevity an advantage? *ACM*, pages 1833–1836.

C. Fellbaum. 1998. *Wordnet, an electronic lexical database*. MIT Press.

Michael Gamon. 2004. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. *Proceedings of the 20th international conference on Computational Linguistics*.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. Technical report, Stanford.

David Haussler. 1999. Convolution kernels on discrete structures. Technical report, University of California at Santa Cruz.

M Hu and B Liu. 2004. Mining and summarizing customer reviews. *KDD*.

S M Kim and E Hovy. 2004. Determining the sentiment of opinions. *Coling*.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, pages 423–430.

Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *Proceedings of the 17th European Conference on Machine Learning*.

Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining. *Proceedings of LREC*.

B. Pang and L. Lee. 2004. A sentimental education: Sentiment analysis using subjectivity analysis using subjectivity summarization based on minimum cuts. *ACL*.

P. Turney. 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. *ACL*.

C M Whissel. 1989. *The dictionary of Affect in Language*. Emotion: theory research and experience, Acad press London.

T. Wilson, J. Wiebe, and P. Hoffman. 2005. Recognizing contextual polarity in phrase level sentiment analysis. *ACL*.