Thesis for the degree of
Master in Language Technology

Sentiment Analysis of Microblogs

Tobias Günther

Supervisor:
Richard Johansson

Examiner:
Prof. Torbjörn Lager

UNIVERSITY OF
GOTHENBURG

June 2013

Contents

1 Introduction

This work examines the problem of sentiment analysis of microblogs, which has become a popular research topic during the last years. The contributions and structure of the thesis are as follows:

First, we give a short introduction to the problem of sentiment analysis in general and motivate sentiment analysis in microblogs, before introducing Twitter, a popular microblogging service on the Internet.

In section 2, we review the previous work on sentiment analysis in Twitter, producing what is to the author's knowledge the most comprehensive overview on studies concerning the topic thus far.

In section 3, we provide a structured survey, summarizing the most common approaches and available resources for addressing the problem of sentiment analysis of microblogs.

In section 4, we conduct experiments to give directions for conflicting results of previous studies by evaluating previously used methods on larger datasets than used before. Furthermore, we propose and evaluate new ideas to enhance classifiers for sentiment analysis of Twitter messages.

In section 5 we discuss our results and give indications for future work.

1.1 Sentiment Analysis

Sentiment analysis, also called opinion mining, is a research area in the field of text mining and natural language processing. The aim of a sentiment analysis system is the automatic detection and classification of moods and opinions towards a so called opinion target. The opinion targets are often named entities, such as individuals, organizations, products, services, but also events and circumstances. In written text, an opinion is usually declared verbally, e.g. by choosing words or phrases expressing sentiment, or non-verbally, e.g. by using emoticons, punctuation or spelling variation. More formally, Liu (2012) defines an opinion as the quintuple $(e_i, a_{ij}, s_{ijkl}, h_k, t_l)$ where "e_i is the name of an entity, a_{ij} is an aspect of e_i, s_{ijkl} is the sentiment on aspect a_{ij} of entity e_i, h_k is the opinion holder, and t_l is the time when the opinion is expressed by h_k. The sentiment s_{ijkl} is positive, negative, or

neutral, or expressed with different strength/intensity levels [...]. When an opinion is on the entity itself as a whole, the special aspect GENERAL is used to denote it. [...] e_i and a_{ij} together represent the opinion target" (Liu, 2012).

Sentiment analysis is usually carried out on one of three different levels. The most common approach is to determine sentiment on document-level, assigning a polarity to the predominant or concluding sentiment in the whole text. Liu (2012) defines this as determining the opinion $(\_, \text{GENERAL}, s, \_, \_)$ of a given document, where "the entity e, opinion holder h, and time of opinion t are assumed known or irrelevant" (Liu, 2012). The second approach is to assign sentiment on the sentence- or clause-level, allowing a more fine-grained analysis of a given document. The most detailed approach however is sentiment analysis on entity- and aspect-level. This task incorporates the detection of entities and their aspects, as well as the linking of a sentiment expression with its target, forming complete quintuples after Liu's definition. Incorporating aspects of entities allows analysis of sentiment towards specific parts of an entity, such the engine of a car, which is a common occurrence in real texts.

Traditionally, sentiment analysis systems are trained and used on longer texts, such as movie or product reviews and blog posts. The problem of sentiment analysis, its challenges and previous work on the topic have been analyzed and summarized many times. For the interested reader we recommend the survey of Pang and Lee (2008) and most recently, the book "Sentiment Analysis and Opinion Mining" by Liu (2012). In this study, we focus on the problem of sentiment analysis in microblogs, which comes with its own peculiarities and challenges.

1.2 Sentiment Analysis of Microblogs

Beside being an interesting research problem, sentiment analysis can be directly applied by persons interested in a large amount of opinions towards a certain topic. This could be for example a private person, who wants to inform herself about the predominant opinion regarding a certain service before purchasing it. Or it could be a company, which wants to analyze customer opinions about their own or their competitors' products to identify possible faults or dissatisfaction, to enhance their products or find new market gaps. Or a political party might be interested in peoples approval or disapproval of their current actions. These are only three examples of real world applications of sentiment analysis, but there are many more imaginable.

With the rise of social media, the number of opinions on the web has multiplied, as platforms like Facebook and Twitter make it very easy for everyone to share their thoughts on literally anything. This calls for sentiment analysis systems that can process large amounts of data and are able to handle the special challenges of the text genre of so-called microblogs. Because of the interest in utilizing this freely available information by research and industry, sentiment analysis of microblogs has become a popular research topic during the last years.

Besides the challenges traditional sentiment analysis systems face, such as ambiguity, handling of negation, detection of sarcasm and opinion spam, sentiment analysis of microblogs have to handle the following additional difficulties:

- **Text Length**: Microblog posts are usually very short. While this can be an advantage, because authors tend to get straight to the point they want to make, it poses the challenge that the expressed opinion might be dependent on one word only. The word might not be available in the used lexical resource or might not have occurred in the training data, which can lead to the loss of the opinion. A discussion of the phenomenon can be found in Bermingham and Smeaton (2010).

- **Spelling variation**: Due to spontaneity, the informal context and length restrictions the spelling in microblog posts tends to have much

greater variability than in other text genres. Phenomena include misspellings, abbreviations (e.g. "gr8" - "great"), emphatic uppercasing ("WHAT THE HELL WAS THAT????"), emphatic lengthening ("The concert was greeeeeeeeeaat!!!") and the use of slang and neologisms. This leads to much more sparsity in the input and is a special challenge for the use of lexical resources. Brody and Diakopoulos (2011) find emphatic lengthening to occur in every 6th tweet of their dataset and provide a detailed analysis.

- **Special tokens**: Tokens uncommon in other text genres, such as URLs and emoticons, can lead to difficulties when trying to use natural language processing tools, such as part-of-speech taggers and syntactical parsers. The latter are often trained on newspaper texts, which are considerably different to microblog posts.

- **Topic variation**: The topics discussed on Twitter are not constrained in any way and the variety is therefore very large. This can cause problems for sentiment analysis, e.g. when words express different sentiment in different contexts.

- **Amount of data**: While the texts as such are often short, the amount of texts can be overwhelmingly large. In 2012 the popular microblogging service Twitter announced[1] 12,233 posts per second about the American football Super Bowl towards the end of the game.

- **Language style**: Due to Twitter's large userbase the variety in writing style is very large. This might range from formal newspaper-like text to very informal slang including profanity. Furthermore, the vocabulary used can change rapidly. All this can lead to problems for annotated training data and lexical resources.

- **Multilingual content**: While online newspapers and blogs tend to be written in one language, users of microblogging platforms use a wide

[1] see `https://twitter.com/twitter/status/166378382660079618`

variety of languages, sometimes even in the same message or sentence. With the shortness of the posts language detection becomes increasingly difficult.

These difficulties do not apply to sentiment analysis exclusively, but are also of concern for other natural language processing tools, such as part-of-speech taggers, parsers and the like.

Twitter

Twitter is currently the most popular microblogging service on the Internet in which most posts are publicly available to everybody. With several hundred million users its userbase is huge and the content produced currently amounts to 58 million tweets per day on average[2]. Twitter also offers an easy to access application programming interface (API), which can be used to interact with the service very easily, e.g. for downloading tweets. For these reasons almost all previous research on sentiment analysis of microblogs has been carried out on Twitter data, which is also the case for this work. Nevertheless, we believe that many findings are transferable to posts of other microblogging services as well. The following list explains some Twitter-specific vocabulary, which will be used throughout the rest of the thesis:

- **Tweet**: This is the name for one post on the Twitter platform. The length of a tweet is restricted to 140 characters.
- **User and username**: To be able to post tweets on Twitter an author has to register with the platform first and is afterwards known under a freely chosen pseudonym. To interact with other users on Twitter, authors can mention other users pseudonyms using the @ symbol (e.g. `@tobigue_`), which leads to the mentioned user being notified about

[2]information from `http://www.statisticbrain.com/twitter-statistics/`, accessed on May 17th, 20:00

the tweet. This is often used in conversations to indicate that a post made is meant as answer to another tweet.

- **Hashtag**: Using the # symbol, users can tag their tweets, indicating the relevance towards a certain topic, e.g. `#SuperBowl`. Those tags can be used by the users to discover other tweets about the same topic. Twitter provides an overview of so-called "trending topics", which are currently discussed by a lot of users.

- **Follower**: Users on Twitter can connect to each other by "following" other people, meaning that they get notified about new tweets by users they follow. Following is not a bidirectional connection such as a Facebook friendship, so every user has separate lists of other users she follows, and users she is followed by.

- **Retweet**: Tweets can be re-distributed by a functionality called retweeting, which used to share a tweet of another user with one's own followers. The tweet is usually unchanged, or only marked with the abbreviation RT to indicate the retweet, often followed by the username of the original author and sometimes a short comment.

2 Related Work

This section provides a detailed overview of previous studies on sentiment analysis of microblogs.

2.1 Studies using supervised machine learning

Go et al. (2009) were among the first to do sentiment analysis specifically on Twitter data. In their paper they treat the problem as one of binary classification, classifying tweets as either positive or negative. Due to the a lack of hand-labeled training data Go et al. (2009) employ distant supervision to train a supervised machine learning classifier: they download a large amount of tweets via the Twitter API and use emoticons in the tweets as noisy labels. Tweets containing emoticons expressing both positive and negative sentiment are not considered. They also remove messages containing retweets and message duplicates. Their final training data set consists of 1,600,000 tweets: 800,000 for each class. They also hand-label a test set of 182 positive and 177 negative tweets. In the data preprocessing step emoticons are removed, as they are used as labels, and generic tokens are inserted for user mentions and links. Furthermore, adjacent repeated letters are collapsed into two letters to reduce the spelling variety introduced through emphatic lengthening to some extent. As features Go et al. (2009) employ unigrams, bigrams, a combination of both and part-of-speech tags. In their experiments they compare the Naive Bayes (NB), Maximum Entropy (MaxEnt) and Support Vector Machine (SVM) classification methods. Their best result is 82.9% accuracy using SVM with only unigrams as features. Adding bigrams results in an increase of the NB and MaxEnt performance, but a decrease in the case of SVM. They report that in their experiments adding negation as an explicit feature and using part-of-speech tags did not improve classification performance, while using bigrams exclusively yields worse results due to the too sparse feature space.

Pak and Paroubek (2010) also use positive and negative emoticons as noisy labels to create their training data of 300,000 tweets. However, they also use the tweets associated with the Twitter accounts of newspapers as

neutral samples, making the problem one of classification with three classes. They remove URLs, usernames, retweets, emoticons and article stopwords (a, an, the) from all tweets and tokenize on whitespace and punctuation. Negations are attached to the preceding and following word. Beside using unigrams, bigrams and trigrams as features, Pak and Paroubek (2010) use part-of-speech tags to compute the posterior probability in their Naive Bayes models. They find these to outperform support vector machines and conditional random fields and report a best result of 0.63 in the rather unpopular $F_{0.5}$ measure. They use the two measures entropy and salience to identify the most informative n-grams and find salience to be the superior measure. Their experiments also confirm the often-made observation that classification performance increases with more training data. Additionally, Pak and Paroubek (2010) provide an analysis of the distribution of part-of-speech tags in tweets of different sentiment.

Barbosa and Feng (2010) build a two step classifier. In the first step tweets are classified as subjective or objective. The subjective tweets are then further classified as positive or negative. They also follow a slightly different approach to create a dataset with noisy labels: they use the judgments of three sentiment detection tools on the Internet. Tweets in which the classification differs between the sources are removed. Additionally, they allow only one sample per Twitter user in the dataset. After this cleaning process, their training data consists of roughly 200,000 tweets for subjectivity detection and 71,046 positive and 79,628 negative tweets for polarity detection. They also manually annotate a 1,000 tweet sized development set and a 1000 tweet sized test set for evaluation. Barbosa and Feng (2010) divide their features into two categories: so-called meta-features and tweet syntax features. The first group holds features such as part-of-speech tags from a part-of-speech dictionary and the prior subjectivity and polarity of words in the MPQA lexicon (see section 3.2). The prior polarity is switched for all words preceded by a negation word and weighted by the occurrence of positive and negative words in the training data. The second group holds more Twitter-specific features such as the presence of retweets, hashtags, URLs, exclamation marks and question marks, emoticons and upper case tokens.

They normalize the frequency of each feature by the number of tokens in the tweet. Altogether they use a total of only 20 features. They get the best results using a SVM classifier for both steps and achieve 81.9% accuracy for the subjectivity detection step, 81.3% accuracy for the polarity detection step, and report a unigram baseline of 72.4% and 79.1%, respectively. They find that the meta-features are more important for the polarity detection step and the tweet syntax features are more significant for subjectivity detection. As no tokens are used directly as features for the classifiers, the authors find that their approach is more robust to bias and noise in the training data and generalizes better in case of little available training data.

Bermingham and Smeaton (2010) specifically investigate the impact of the shortness of tweets on sentiment analysis. They collect tweets of ten so-called trending topics for each of the five categories "entertainment, products and services, sport, current affairs and companies" (Bermingham and Smeaton, 2010) to build a manually annotated dataset of 1,410 positive, 1,040 negative and 2,597 neutral tweets. In their experiments, the authors compare the results of classifying tweets, blog posts, movie reviews and microreviews. In the preprocessing of tweets they replace topic words, URLs and usernames with generic placeholders. As features for their machine learning classifiers unigrams, bigrams and trigrams as well as part-of-speech tags and part-of-speech n-grams are used. They find the Naive Bayes classifier to outperform support vector machines on microblog data, but not on longer texts such as blogs. They report their best result for binary positive/negative classification as 74.85% accuracy and 61.3% for the ternary case, both using Naive Bayes and unigrams. Using n-grams and part-of-speech tags increased the classification performance only in the case of longer texts (blogs and movie reviews). Bermingham and Smeaton (2010) conclude that sentiment analysis of tweets is an easier task than sentiment analysis of longer texts. They also find boolean feature vectors to perform better than frequency based ones. While they identify part-of-speech stopworded bigrams to be useful features for support vector machines, they did not observe any improvements by using part-of-speech n-grams, stemming or stopwording.

Bifet and Frank (2010) address the challenges of the large size of Twit-

ter data streams. They propose a new kappa-based sliding window measure for evaluating classification performance in data streams, which can handle changes in the data distribution over time, which is a likely possibility when working with the rapidly changing content of social media. They experiment with the Stanford Twitter Sentiment dataset of Go et al. (2009) and the Edinburgh Twitter Corpus of Petrovic et al. (2010), using emoticons as noisy labels. After replacing usernames and URLs with generic placeholders and collapsing repeated adjacent letters to a maximum of two, they use only unigrams as features. Besides their evaluation in the newly proposed metric, the authors report 82.45% accuracy on the test set of the first corpus using Naive Bayes and 86.26% accuracy on the second corpus using stochastic gradient descent (SGD) as best results for the binary classification task. While they find Naive Bayes to rival SGD, they report that classification using a Hoeffding tree yields inferior results and advise against using tree learners in the context of large data streams. Instead, they recommend SGD as learning method in this setting as it can adapt to changes over time and the change of the feature weights can be used to monitor the change in sentiment towards certain topics in a microblog data stream.

Davidov et al. (2010) use 50 hashtags and 15 emoticons as noisy labels to label the dataset of O'Connor et al. (2010). They use words, n-grams (2-5), tweet length, punctuation and numbers of exclamation marks, question marks, quotes and capitalized/all caps words in the sentence as features. Additionally, they identify special patterns of high-frequency words and content words and use those as features as well. As best result for their k-nearest neighbor like classification strategy they report an average harmonic F-score of 86.0% for binary classification. They also try to classify the tweets into the classes introduced by the 50 hashtags, which expectedly yield much lower results. The authors find words, patterns and punctuation features to be useful, while n-grams increase classification performance only marginally, despite their strategy of only including tokens that surpass a 0.5% frequency threshold in the n-grams in the training data.

In contrast to most other studies, **Agarwal et al. (2011)** do not filter out tweets in another language than English when collecting their training set

with the Twitter API, but use Google Translate for translation. They create a dataset of 11,875 manually annotated tweets. After 3,122 tweets were disregarded, mostly because of errors in the automatic translation, the remaining 8,753 tweets are labeled as positive, neutral or negative. For their experiment they use a balanced subset of this data, containing 1,709 tweets of each class. They label 170 emoticons listed in Wikipedia with the five categories extremely-positive, extremely-negative, positive, negative, and neutral and substitute the occurrence of the emoticons in the tweets with their sentiment annotation. Using a lexicon holding 5,184 entries, popular acronyms are replaced with their corresponding long forms. Furthermore, usernames, URLs and negations are replaced by generic placeholders and repeated sequences of the same character are reduced to a number of two. Prior word polarities are computed using the Dictionary of Affect in Language (DAL) by Whissell (1989). Words not found in the dictionary are assigned the prior polarity of their WordNet (Fellbaum, 1998) synonyms present in the DAL, if available. Beside unigrams Agarwal et al. (2011) use 50 additional features, capturing counts, sums and percentages of prior polarities and occurrences of a diverse range of tokens and other phenomena such as capitalized text. Additionally, those features are calculated again for the last third of the tweet. In their experiments the authors compare five different models, all based on support vector machine classifiers. Similar to Barbosa and Feng (2010) the authors find that a classifier using only abstract linguistic features performs as well as the unigram baseline. For the binary positive/negative classification task they report the best result to be 75.39% accuracy using a model combining unigrams and the abstract linguistic features. For the three-way task including the neutral class the best performing model combines the abstract linguistic features with a special tree representation of the token, which is used with a SVM partial tree kernel (Moschitti, 2006) and yields an accuracy of 60.83%. While both results lie about 4% above their respective unigram baseline using all data, the advantage of the more sophisticated methods is even larger when using a smaller training set. While Agarwal et al. (2011) can find improvement using part-of-speech tag based features and report the most informative abstract linguistic features to be the ones combining the

prior polarity of words with their part-of-speech tags, they see only marginal performance improvements by adding Twitter-specific features such as hashtags and URLs.

Aisopos et al. (2011) propose a language independent model for sentiment classification in Twitter based merely on character n-grams. Their training data consists of 1 million tweets for each of the three classes positive, negative and neutral, annotated with noisy labels using the presence or absence of emoticons. While they can report an improvement of about 4% in accuracy for the binary classification task and about 6% for the three-way problem over a tf-idf term baseline, they do not compare to the often better performing baseline using boolean unigram features. Their best model is based on a 4-gram graph with distance-weighted edges and the C4.5 tree-learner and achieves 66.77% accuracy with two classes and 50.67% in case of three classes.

Jiang et al. (2011) apply a three step approach to target-dependent sentiment analysis of tweets. Like Barbosa and Feng (2010) they first classify tweets as subjective or objective and decide positivity or negativity with a separate classifier. The third, novel step of their approach is a graph-based optimization method using related tweets, such as tweets about the same opinion target by the same author or replies to a tweet. Their manually labeled dataset consists of 1212 neutral, 459 positive and 268 negative tweets towards the five opinion targets "Obama", "Google", "IPad", "Lakers" and "Lady Gaga". The authors employ rule-based normalization and stemming with a dictionary of 20,000 entries for preprocessing. To identify opinion targets they make use of the OpenNLP[3] part-of-speech tagger and use the maximum spanning tree dependency parser by McDonald et al. (2005) to create target-dependent features and handle negation. To increase coverage, they additionally apply co-reference resolution following Soon et al. (2001) and find related noun phrases using pointwise mutual information (PMI). Additionally, unigrams, punctuation, hashtags, emoticons and the

[3]see `http://opennlp.apache.org/`

the General Inquirer (Stone et al., 1966) lexicon are used to generate target-independent features. In their evaluation Jiang et al. (2011) find that the presented method outperforms a reimplementation of Barbosa and Feng (2010) in subjectivity classification (7.9% higher) as well as polarity classification (1.7% higher). They report an accuracy of 68.3% for the overall three-class classification task, which is lowered to 66% when not using the graph-based optimization.

Kouloumpis et al. (2011) use frequent hashtags indicative of sentiment as noisy labels for tweets from the Edinburgh Twitter Corpus by Petrovic et al. (2010). Furthermore, they conduct experiments on the Stanford Twitter Sentiment corpus labeled using emoticons by Go et al. (2009) and a hand-annotated set of 4000 tweets for evaluation. In the preprocessing step they replace abbreviations using various lexical resources on the web, normalize spelling after noting emphatic uppercasing and lengthening and insert generic placeholders for hashtags, usernames and URLs. As features they use the top 1000 stopword removed uni- and bigrams considering their information gain measured by chi-squared, count of part-of-speech tags, presence or absence of Twitter specific tokens and prior word polarities according to the MPQA subjectivity lexicon. They find that an AdaBoost learner outperforms support vector machines and report their best performance as 75% accuracy for the three class task using the combined training corpora on the evaluation data and all features but counts of part-of-speech tags, which they find to lower performance.

Saif et al. (2011) investigate two different approaches of so-called semantic smoothing to address the problem of data sparseness by extracting hidden semantic concepts. They use a balanced subset of 60,000 tweets from the dataset of Go et al. (2009) as well as the corresponding test set of 177 negative and 182 positive manually labeled tweets. As method of classification they employ a Naive Bayes classifier trained with unigram features. To extract the hidden concepts they use a third party service called Alche-

myAPI[4] to identify named entities in the tweets. In the first approach they attempt shallow semantic smoothing by simply replacing the found named entities with their category. In the second approach they interpolate the Naive Bayes classifier with a generative semantic model. While they find the first approach lowers classification performance compared to a unigram Naive Bayes classifier with no replacement, they report the second approach slightly outperforms the baseline with a result of 81.3% accuracy for the binary positive/negative classification task. These approaches are developed further in Saif et al. (2012a) in which they add Twitter specific preprocessing like replacement of usernames and URLs with generic placeholders and collapsing of repeated adjacent letters to a maximum of two. Furthermore, hashtags, single characters and digits and non-alphanumeric characters are removed. They enlarge the test set of Go et al. (2009) to 1000 tweets and can report 84% accuracy with the enhanced interpolation approach, which is an increase of about 3% compared to the baseline. Another approach called joint sentiment topic model (originally introduced by Lin and He (2009)) is tested in the paper as well and yields an accuracy of 82.3%. The authors test their approaches on two additional datasets (Health Care Reform by Speriosu et al. (2011) and Obama McCain Debate by Shamma et al. (2009)) in Saif et al. (2012b) and come to the conclusion that semantic smoothing works best on large datasets with many topics, while the sentiment topic models are a better choice for small datasets with specific topics.

Liu et al. (2012) propose an approach called emoticon smoothed language model (ESLAM) which is able to integrate data labeled with noisy labels with manually annotated tweets. The authors first train a language model for 570 positive, 654 negative and 2503 neutral tweets of the Sanders corpus[5]. Then they use noisy emoticon data collected via the Twitter API to smooth those language models. For classification they pick the class of the model yielding the highest score. In the preprocessing step they insert generic placeholders for usernames, digits and URLs, remove stopwords, con-

[4]see `http://www.alchemyapi.com/`
[5]see `http://www.sananalytics.com/lab/twitter-sentiment/`

duct stemming and lowercasing and remove retweets and duplicates from the dataset. Additionally, they distinguish links pointing to popular picture and video websites from other links. They find that their emoticon smoothed language model performs better than using a fully supervised language model and report an accuracy of 82.5% as their best result for polarity classification and 79.5% for subjectivity classification. In an evaluation of the effect of using manually labeled data the authors find that it significantly improves classification performance compared to using only data labeled with noisy labels.

As part of this thesis, in **Günther and Furrer (2013, in press)** we present a system submitted to the SemEval-2013 shared task on sentiment analysis in Twitter (Nakov et al., 2013, in press). The data used for the subtask of identifying the predominant document-level sentiment consists of 3855 positive, 4889 neutral and 1624 negative manually labeled tweets. In the preprocessing step we apply normalization techniques such as lowercasing, substitution of digits and collapsing of repeated letters to address the variability in spelling introduced by emphatic lengthening. Normalized tokens and word stems are used as unigram features. Furthermore, we compute a prior tweet polarity using SentiWordNet (Baccianella et al., 2010) and use presence or absence of the token in Twitter word clusters computed by Owoputi et al. (2013) from 56,345,753 tweets using Brown clustering (Brown et al., 1992). For classification we find a linear model trained with stochastic gradient descent, hinge loss and elastic net regularization to outperform other methods and observe that the choice of the training algorithm can become even more important than the choice of features with increasing training set size. While we find all features to contribute to the final model, the Twitter word cluster features cause the highest loss in performance when removed. In the evaluation on a hidden test set of 3813 tweets we were able to reach a F_1 of 65.27%[6], which was announced to be the 2nd best performance of 52 competing systems.

[6]despite being a classification problem with three classes, the shared task was evaluated on the average F_1 of the positive and negative class

2.2 Studies using lexicon-driven methods

O'Connor et al. (2010) use the MPQA sentiment lexicon of Wiebe et al. (2005) to identify sentiment in tweets mentioning Barack Obama. To classify a tweet they simply count if it contains more positive or negative words according to the sentiment lexicon. Even though this is a very simple approach, they report significant correlation between the aggregated sentiment in tweets and public opinion polls published by Gallup.

Marchetti-Bowick and Chambers (2012) also use tweets for political forecasting and compare a purely lexicon-based approach to a data-driven approach with noisy labels. They find that the latter strongly correlates with Gallup polls, while the former does not.

Thelwall et al. (2010) propose a lexicon-based algorithm called SentiStrength, which assigns a polarity (positive/negative) and corresponding strength value between 1 and 5 to a given text. Beside their list of 298 positive and 465 negative terms annotated with polarity and strength values, SentiStrength uses lists of emoticons, negations and boosting words in the decision process. To deal with emphatic lengthening the authors propose a three step method to reduce words to their standard form. When comparing SentiStrength to various machine learning classifiers on MySpace comments, the authors find their method to perform better for classifying negative sentiment, but not for positive sentiment. The algorithm is enhanced in **Thelwall et al. (2012)**, where the authors introduce idiom lists and strength boosting by emphatic lengthening as well as an unsupervised version of SentiStrength. Furthermore, the number of terms in their sentiment strength wordlist is increased from 693 to 2310. They again compare their algorithm with different machine learning algorithms on six different datasets, including Twitter data, and find especially logistic regression to outperform SentiStrength.

Zhang et al. (2011) develop a rule-based approach for entity-level sentiment analysis in Twitter. They compute a sentiment score for each entity based on its textual proximity to words from a sentiment lexicon. They also perform simple anaphora resolution by resolving pronouns to the closest entity in the tweet. The rule-based algorithm differentiates between declar-

ative, imperative and interrogative sentences and can, among other things, handle comparative sentences, negation and but-clauses. To enhance the recall of the proposed methods, the authors identify additional tweets that are likely to be opinionated and train a support vector machine to assign polarity labels to the contained entities.

Kumar and Sebastian (2012) propose a method to calculate a sentiment score for a tweet based on a sentiment lexicon, which additionally takes emoticons, emphatic lengthening, uppercasing and other clues like the occurrence of exclamation marks in the tweet into account.

Maynard et al. (2012) discuss the special challenges of sentiment analysis in the domain of social media messages and design a rule-based approach building on a shallow linguistic analysis including named entity extraction and event recognition to produce a polarity and score for a given tweet.

Nielsen (2011) describes the process and evaluation of building a new sentiment word list from Twitter data.

Minocha and Singh (2012) also investigate how to build a sentiment lexicon from Twitter data and find value in creating domain-specific sentiment lexica.

2.3 Studies using graph-based label propagation

Speriosu et al. (2011) propose a label propagation method that leverages a graph of tweets incorporating unigrams, bigrams, hashtags, emoticons and users as nodes. They compare their approach to a simple lexicon-based method and a maximum entropy classifier on three different datasets: The Stanford Twitter Sentiment dataset by Go et al. (2009), the Obama McCain debate dataset of Shamma et al. (2009) and their own dataset, which provides 1418 manually annotated tweets related to a discussion about a health care reform in the USA in 2010. While this dataset also holds neutral tweets and opinion targets, the authors only use the positive and negative tweets for this study. Tokenization is done by splitting on whitespace, all characters are converted to lowercase and non-alphanumeric characters are removed from the beginning and end of each token before stopword-removed unigrams and stopword including bigrams are extracted as features. This is based on the assumption that stopwords alone express no sentiment, but can be important in cases like "shit" (negative) vs. "the shit" (positive). The authors find their label propagation method to outperform the other two methods and report 84.7% accuracy on the Stanford Twitter Sentiment dataset, 71.2% on the health care reform dataset and 66.7% on the Obama McCain debate dataset.

Cui et al. (2011) also experiment with a label propagation method and compare its performance to popular online tools for sentiment analysis of tweets and a dictionary based approach. They investigate the special role of emoticons in tweets and also use a graph propagation algorithm to classify the sentiment of tweets in other languages than English.

Wang et al. (2011) conduct experiments on classifying the sentiment of certain hashtags and propose a graph-based model that incorporates co-occurrence of hashtags in tweets. They compare different algorithms to a two-step support vector machine trained with unigrams, punctuation and emoticons (55.96% accuracy) and find an algorithm called loopy belief propagation to outperform the other tested methods (77.72% accuracy).

3 Survey

This section summarizes the previous research of sentiment analysis in Twitter in a structured way and can be used as an overview of common practices for the interested reader new to the field. For a more detailed review of previous work please refer to section 2.

3.1 Data

Getting Twitter data is comparably easy, as Twitter offers an easy to use API.[7] Restricted by an hourly limit of API calls, all tweets can be accessed via different HTTP endpoints, such as tweets posted by a specific user, tweets containing specific terms or by the individual ID of each tweet. The tweets and their associated meta information such as the date, author, language, location, timezone and more are returned in JSON format, which is a popular format for data exchange and is supported by many programming languages.[8]

For creating new datasets the so-called streaming API is most interesting, as it provides unlimited access to the live stream of incoming tweets matching a given query. Typical queries are either lists of emoticons, to create annotated datasets with noisy labels, frequent words such as stopwords, to capture a mostly unbiased stream of tweets with a wide variety of different topics or specific terms like names of entities, to create datasets concerning a certain topic. Also hashtags can be useful as query, as they can be seen as some sort of label the author attaches to the message.

Even though the Twitter API makes it easy to download data from the platform, there are difficulties for scientific research. Since a change of their terms of service in March 2011, Twitter no longer allows the redistribution of collected tweets, such as in annotated datasets. It is allowed to share tweet IDs, which can be used to download the corresponding tweets via the Twitter API, however Twitter users can choose to delete or privatize their tweets. This means that annotated datasets become partly inaccessible over

[7] see `https://dev.twitter.com/docs`

[8] see `http://www.json.org` and `http://en.wikipedia.org/wiki/JSON`

time, which is a serious problem for the comparability of reported evaluation results and the approval of resources required to create new annotations. As of today the only option seems to be to privately share datasets between researchers.

Annotation

By far the most popular way of annotation of datasets for sentiment analysis in Twitter is treating the problem as document-level sentiment classification task and categorizing each tweet with one label. The categories used are in most cases "positive" and "negative" sentiment, in recent studies often complemented by a third category "neutral". Only very few studies employed more fine-grained scales, such as intensity levels from -5 to +5, or more differentiated target-levels such as sentence-, clause- or entity-level.

Manually annotated datasets are often created using human judges on the Amazon Mechanical Turk platform[9] and are usually between several hundred and a few thousand tweets. When reported, the kappa coefficient of inter-annotator agreement ranges from 0.55 to 0.94 with two categories and 0.43 to 0.72 in the case of three.[10] Several studies have used noisy (or "weak") labels to train classifiers by so-called distant supervision. In these cases large amounts of tweets were downloaded via the Twitter API and labeled automatically using simple rules, e.g. utilizing the presence of either positive or negative emoticons in the tweets. Datasets annotated in this manner often hold several hundred thousands tweets. Even though researchers report encouraging results using data with noisy labels, Liu et al. (2012) find that manually labeled data leads to superior results.

Wiebe et al. (2005) provide a general study on "annotating expressions of opinions and emotions in language".

[9] see `https://www.mturk.com/mturk/`

[10] While Cohen's kappa (Cohen et al., 1960) can only be used for two annotators, Fleiss' kappa (Fleiss et al., 1981) is used to measure inter-annotator agreement between more than two judges.

Existing Datasets

The following datasets have been used in more than one study on sentiment analysis in Twitter:

- **Stanford Twitter Sentiment (STS)**[11] (Go et al., 2009):
  - 1,600,000 tweets (800,000 positive, 800,000 negative)
  - Automatic annotation by using the occurrence of positive and negative emoticons (noisy labels)
  - Collected between April 6 and June 25, 2009 by selecting tweets containing `:)` or `:(`
  - Additional 182 positive and 177 negative manually annotated tweets for testing
- **Health Care Reform (HCR)**[12] (Speriosu et al., 2011):
  - 2525 tweets (541 positive, 470 neutral, 1381 negative, 79 irrelevant, 44 unsure)
  - Manually annotated with respect to the opinion targets "conservatives", "dems", "gop", "hcr", "liberals", "obama", "other", "stupak", "teaparty"
  - Collected in March 2010 by selecting tweets containing the #hcr hashtag
- **Obama McCain Debate (OMD)**[13] (Shamma et al., 2009):
  - 3238 tweets (positive, negative, mixed, other)
  - Each tweet is manually annotated by up to 8 annotators using the Amazon Mechanical Turk platform
  - Collected during the presidential debate in September 2008

[11] available at `http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip`
[12] available at `https://github.com/downloads/utcompling/Scalabha/scalabha-0.2.5.zip`
[13] available at `http://www.infochimps.com/datasets/twitter-sentiment-dataset-2008-debates`

- **SemEval-2013 Task 2 Subtask B (SEM)**[14] (Nakov et al., 2013, in press):
  - 15196 tweets (5810 positive, 6979 objective/neutral, 2407 negative)
  - Manually annotated using the Amazon Mechanical Turk platform
  - Collected by selecting tweets that hold named entities and words present in SentiWordNet
- **Sanders Corpus (SAN)**[15]:
  - 5513 tweets (570 positive, 2503 neutral, 654 negative, 1786 irrelevant)
  - Manually annotated with respect to the opinion targets "Apple", "Google", "Microsoft", "Twitter"
- **Edinburgh Twitter Corpus (ETC)** (Petrovic et al., 2010):
  - 96,369,326 tweets
  - Not annotated for sentiment in original paper, but popular Twitter corpus in other studies
  - Bifet and Frank (2010) use emoticons as noisy labels and get 324,917 negatively and 1,813,705 positively annotated tweets
  - Collected between November 11, 2009 and February 1, 2010 from the whole Twitter stream
- **O'Connor et al. Twitter Corpus (OTC)** (O'Connor et al., 2010):
  - 1,000,000,000 tweets
  - Not annotated for sentiment
  - Davidov et al. (2010) use hashtags and emoticons as noisy labels
  - Collected in 2008 and 2009

For results on the datasets in previous work see Tables 1 and 2 in Section 3.6.

[14] available at `http://www.cs.york.ac.uk/semeval-2013/task2/index.php?id=data`
[15] available at `http://www.sananalytics.com/lab/twitter-sentiment/`

3.2 Resources

The following list gives an overview over lexical resources and tools which are specific to sentiment analysis and/or Twitter:

Lexical resources

- **MPQA subjectivity lexicon**[16] (Wiebe et al., 2005), defining prior polarities for over 8000 words
- **SentiWordNet**[17] (Baccianella et al., 2010), assigning sentiment scores to WordNet (Miller, 1995) synsets
- **Opinion Lexicon by Bing Liu**[18], a list holding around 6800 positive or negative words
- **General Inquirer**[19] (Stone et al., 1966), a lexicon holding information on 11,896 words including sentiment values
- **Twitrratr wordlist**[20], consisting of 174 positive and 185 negative words
- **Twitter Word Clusters**[21] (Owoputi et al., 2013), word clusters obtained by Brown Clustering (Brown et al., 1992) of 56,345,753 tweets

NLP Tools

- **Tokenizer and part-of-speech tagger for Twitter**[22] (Owoputi et al., 2013)
- **Named Entity Recognition for Twitter**[23] (Ritter et al., 2011)

[16] available at `http://mpqa.cs.pitt.edu/`
[17] available at `http://sentiwordnet.isti.cnr.it/`
[18] available at `http://www.cs.uic.edu/ liub/FBS/sentiment-analysis.html#lexicon`
[19] available at `http://www.wjh.harvard.edu/ inquirer/`
[20] see Appendix A
[21] available at `http://www.ark.cs.cmu.edu/TweetNLP/`
[22] available at `http://www.ark.cs.cmu.edu/TweetNLP/`
[23] available at `https://github.com/aritter/twitter_nlp`

3.3 Preprocessing

Due to the specific language of tweets and its challenges discussed in section 1.2, adequate preprocessing is an important part of sentiment analysis systems for microblogs. The following gives an overview of common strategies:

- Insertion of generic **placeholders** for Twitter specific tokens such as usernames and URLs. Sometimes different tokens are used for URLs pointing to popular websites for sharing images and videos and other websites.
- Token **normalization**, such as lowercasing and replacement of numbers
- **Collapsing** of repeated adjacent letters to a maximum of two to reduce vocabulary size (e.g.: coooool → cool, nooooo → noo)
- **Stemming** algorithms such as Porter stemming (Porter, 1980) have been used to reduce morphological variety
- Substitution of common **abbreviations** by their long forms
- Handling **negation** is important for sentiment analysis, as negation words can switch the polarity of a sentiment expression (e.g. good → not good). For sentiment analysis in tweets only simple strategies, like attaching a negation marker to the adjacent tokens, have been used. Prior polarities from a subjectivity lexicon are often switched if they occur after a negation expression.
- Several studies **remove certain tweets**, such as retweets and duplicates, to get a more balanced dataset. Some also choose to allow only one tweet per user in the dataset for the same reason.
- If the dataset was labeled using noisy labels, the tokens used as labels (usually emoticons) are often removed to **reduce the introduced bias**. The same is done for words used as queries to collect tweets about a certain topic.

3.4 Features

This section gives an overview over features used in studies on sentiment analysis in Twitter:

- **Unigrams**: As previous research has found good results for automatic sentiment analysis using a simple bag of words model (Pang et al., 2002), this approach was also adapted for sentiment analysis in microblogs. A classifier trained on unigrams only is often used as baseline to show the effectiveness of a newly proposed method. Many studies report a simple boolean presence/absence strategy to yield better results than using more advanced weighting schemes such as tf-idf.

- **N-grams**: Some studies have tried using bi- and trigrams to capture more context-dependent meanings in tweets. However, at this time it is unclear if using bi- or other n-grams always leads to improvement of classification performance. If used, higher n-grams should only be used in addition to unigrams, as using only e.g. bigrams leads to a too sparse feature space. To address this problem, part-of-speech tag filtered n-grams can be used (see Matsumoto et al., 2005).

- **Sentiment wordlists**: Manually compiled lists of words expressing sentiment have been used as the only resource in the so-called lexicon approach, but also as features in machine learning classifiers. If a list provides polarity values on a numeric scale, a common means of feature extraction is to compute the average polarity value of all words in a tweet present in the list.

- **Part-of-speech tags**: Providing information on a higher level of abstraction, part-of-speech tags have been a popular choice of researchers to introduce features that can capture information from tokens outside of the vocabulary of the training data. As for n-grams, not all studies using part-of-speech tags as features for sentiment analysis of tweets have been able to report an actual improvement of classification performance by using them. Pak and Paroubek (2010) provide an in-depth

analysis of the distribution of part-of-speech tags in tweets of different sentiment.

- **Occurrence of specific tokens**: Several studies have used features indicating the presence or absence of specific token, such as exclamation marks, questions marks, quotes, hashtags, usernames, URLs and emoticons.

- **Occurrence of special spelling**: Non-standard ways of spelling have been employed as sentiment indicators in different studies. Features capturing repeated punctuation, emphatic lengthening and emphatic uppercasing have been used.

- **Dependency trees**: Dependency parsers have been used to produce syntactic annotation of tweets. This is especially important for entity-level sentiment analysis, as a syntactical relation is one of the best indicators for a connection of the word expressing the sentiment and the opinion target. Sometimes anaphora and coreference resolution is used to increase coverage.

- **Word clusters**: Clusters of words occurring in similar contexts have been used to address the problem of sparseness in the training data. For example clusters produced by the Brown clustering algorithm (Brown et al., 1992) can be useful for Twitter data, as spelling variants of the same words and semantically related but new words often end up in the same cluster.

- **Twitter graph**: Connections between tweets and users on the Twitter platform have been used to enhance sentiment classification performance, leveraging e.g. other tweets of an author, retweets, answers to tweets, tweets of users connected to an author and so forth.

3.5 Methods

There are three main approaches in the previous research on sentiment analysis on Twitter. As with sentiment analysis in general, the most popular approach is to train supervised machine learning classifiers, either on manually annotated data or data labeled by "noisy" labels such as emoticons. Many authors report their best results using support vector machines or a Naive Bayes classifier. The second approach is lexicon-based methods, which assign labels or scores to tweets by aggregating scores found in sentiment lexica. Sometimes the aggregation simply consists of summing or averaging over the polarity values of all words of the tweet found in the used lexicon, while in other cases rules are used to address linguistic circumstances such as negation. The third approach is based on label propagation methods, leveraging the connections between tweets and users on the Twitter platform to assign sentiment values to closely connected tweets. While most research can be categorized as one of these approaches, some studies also combine them.

3.6 Results

The most popular way of reporting results in previous research is by the measure of accuracy. When using unbalanced datasets it is advisable to also evaluate the F-measure, which has been used for evaluation by several studies beside accuracy. Bifet and Frank (2010) propose a new kappa-based measure especially designed for large and changing data streams such as Twitter data.

As expected, studies addressing the classification problem of positive vs. negative sentiment report higher performance than studies dealing with the harder problem of classifying positive vs. neutral vs. negative sentiment. Tables 1 and 2 give an overview of the reported results in the reviewed literature, only considering the best reported result for each paper.

Author	**Result**	**Measure**	**Corpus**
Speriosu et al. (2011)	84.7	Acc	STS
Saif et al. (2012a)	84.0	Acc	STS+own
Go et al. (2009)	82.9	Acc	STS
Bifet and Frank (2010)	82.45	Acc	STS
Saif et al. (2012b)	83.9	F_1	STS
Saif et al. (2011)	82.3	Acc	STS
Speriosu et al. (2011)	71.2	Acc	HCR
Saif et al. (2012b)	68.15	F_1	HCR
Saif et al. (2012b)	78.2	F_1	OMD
Speriosu et al. (2011)	66.7	Acc	OMD
Bifet and Frank (2010)	86.2	Acc	ETC
Davidov et al. (2010)	86.0	F_1	OTC
Agarwal et al. (2011)	75.39	Acc	own
Bermingham and Smeaton (2010)	74.85	Acc	own
Aisopos et al. (2011)	66.77	Acc	own
Pak and Paroubek (2010)	0.63	$F_{0.5}$	own

Table 1: Results of studies addressing 2-way classification

Author	**Result**	**Measure**	**Corpus**
Liu et al. (2012)	65.5*	Acc	SAN
Kouloumpis et al. (2011)	0.75	Acc	own
Jiang et al. (2011)	68.3	Acc	own
Barbosa and Feng (2010)	66.5*	Acc	own
Bermingham and Smeaton (2010)	61.3	Acc	own
Agarwal et al. (2011)	60.83	Acc	own
Aisopos et al. (2011)	50.67	Acc	own

Table 2: Results of studies addressing 3-way classification.

* Results marked with stars were reported as two separate results for the subjectivity and the polarity classification step, which were multiplied, assuming the authors did not count errors twice.

4 Experiments

The experiments were conducted using the Python programming language and the open source machine learning toolkit *scikit-learn*[24], which provides fast implementations for a wide range of different machine learning algorithms (Pedregosa et al., 2011). Table 3 provides an overview of the data used in the following experiments. Further information about these datasets and other resources used can be found in Section 3.1. We dismiss all tweets with annotations that are not positive, neutral or negative and in case of the OMD dataset we only use those tweets having a two-third majority vote from the annotators.

Dataset	**Positive**	**Neutral**	**Negative**	**Total**
HCR	541	470	1,381	2,392
OMD	709	-	1,195	1,904
SAN	498	2,230	549	3,277
SEM	5,427	3,732	2,225	11,384
ALL	7,175	9,299	5,350	21,754
STS_{test}	192	-	177	369

Table 3: Size of the datasets used in the experiments in tweets.

We use two different experimental setups in the experiments: 10-fold cross validation and evaluation on a dedicated test set. In 10-fold cross validation, the complete data set is split into 10 parts, the classifier is trained on 9 parts and 1 part is used as test set for evaluation. This is done 10 times, so that every part is used once for evaluation. We use stratified splits, meaning that the proportion of the classes in one split is similar to the proportion of the classes in the whole training set. Cross validation is used to account for inhomogeneity in the training data and has been found to yield more stable and meaningful results compared to testing on a static training set / test set split. Nevertheless, for comparison with other work testing on a dedicated test set can be useful. In the following experiments we also use

[24]see `http://scikit-learn.org/`

the HCR, OMD, SAN and SEM datasets combined - this is referred to as ALL and constitutes so far the largest manually annotated dataset used in Twitter sentiment analysis known to the author. We decide not to balance the dataset between classes, hoping to make our findings more relevant to real world applications, where data is probably not balanced either. The STS_{test} dataset is kept as a separate test set and is not used in the experiments, to allow an unbiased comparison of the final model with previous work using the same dataset for evaluation.

4.1 Baseline

As in previous research, we use whitespace tokenized unigrams as boolean features for our baseline. Machine learning algorithms trained on unigrams have shown to perform reasonably well for sentiment analysis tasks (Pang et al., 2002) and have been used as baselines in other studies on sentiment analysis of Twitter data. Table 4 shows the baseline performance for the multinomial Naive Bayes (MNB) and support vector machine (SVM) classifiers often used in previous studies for each dataset.[25] Additionally, the table contains the majority baseline (MAJ), showing the performance of a classifier always picking the most frequent class in the dataset.

In the results we can observe that in almost all cases the support vector machine outperforms the Naive Bayes classifier. As expected, the results on the 2-class classification problem are considerably higher than for the 3-class task and are mostly in the same ranges as previous results on these datasets.

4.2 Preprocessing

In previous studies different preprocessing techniques have been used, but never compared. In the following we conduct a series of experiments to evaluate different preprocessing methods. All experiments in this section

[25] For the baseline experiments we used the `MultinomialNB()` and `SGDClassifier(loss='hinge', penalty='l2', alpha=0.001, n_iter=100, shuffle=True, random_state=0)` classes from *scikit-learn*

	MNB		SVM		MAJ	
Dataset	Acc	F1	Acc	F1	Acc	F1
HCR_2	75.03	68.21	**76.17**	**68.97**	71.85	41.81
OMD_2	79.57	77.79	**80.57**	**78.67**	62.76	38.56
SAN_2	**81.57**	**81.41**	78.80	78.74	52.44	34.40
SEM_2	78.99	72.91	**81.38**	**75.83**	70.92	41.49
ALL_2	79.11	78.57	**80.30**	**79.76**	57.29	36.42
HCR_3	65.09	55.87	**65.76**	**58.50**	57.73	24.40
SAN_3	70.61	57.02	**77.33**	**65.26**	68.05	27.00
SEM_3	62.44	54.34	**67.46**	**61.84**	46.04	21.02
ALL_3	65.63	65.25	**68.87**	**67.47**	42.42	19.86

Table 4: Baselines: 10-fold cross-validation performance of unigram classifiers on different datasets. The subscript indicates the numbers of classes used (2=positive/negative, 3=positive/neutral/negative) and the best result for each dataset and measure is marked in bold.

were conducted using 10-fold cross validation on the ALL_3 dataset.

Tokenization

We compare four different methods of tokenization:

1. **WSP**: Splitting on whitespace
2. **RE1**: A regular expression that matches alphanumeric character sequences: `[a-zA-Z0-9]+`
3. **RE2**: A regular expression that matches either URLs, alphanumeric character sequences (plus apostrophe) or non-alphanumeric non-whitespace character sequences[26]:
   `https?://[\S]+|[[^\W\_]|['’]]+|[[\W\_]--[\s]]+`
4. **TOK**: Using the rule-based Twitter specific tokenizer of Owoputi et al. (2013)

[26]we use the `regex` package for Python (`https://pypi.python.org/pypi/regex`), because in contrast to the `re` package of the standard library it allows set operations.

	WSP	RE1	RE2	TOK
MNB	65.25	63.65	65.51	65.96
SVM	67.47	68.01	69.46	**69.57**

Table 5: Performance of baseline classifiers with different tokenization methods, measured in average F_1.

Table 5 shows the results of applying different tokenization methods prior to training the baseline classifiers. We can observe that only using alphanumeric tokens (RE1) yields the worst results, indicating that punctuation can provide valuable information for sentiment analysis in Twitter. While using a regular expression that splits punctuation sequences, such as emoticons, from words (RE2) yields a slight improvement compared to just splitting on whitespace (WSP), the best result is achieved when using the Twitter specific tokenizer (TOK).

As a second method of evaluation we compute how many types (unique lowercased tokens) of the ALL_3 dataset are present in three popular sentiment lexica (see Section 3.2). Table 6 shows the results.

	WSP	RE1	RE2	TOK
MPQA	2,264	**2,502**	2,492	2,468
LIU	1,992	**2,227**	2,216	2,197
SWN	10,395	**11,778**	11,709	11,429
#types	73717	46218	49549	53125

Table 6: Number of types in the ALL_3 dataset present in three popular sentiment lexica, dependent on the tokenization method.

Interestingly, the simple extraction of alphanumeric character sequences (RE1) can achieve a higher lexical coverage than any other method. This is a useful information for purely lexicon-based methods and for machine learning classifiers incorporating lexical resources in their features. In the following experiments we will use the tokenizer by Owoputi et al. (2013) and come back to the other methods if needed.

Normalization

To address emphatic uppercasing ("That's GREAT!!") and emphatic lengthening ("cooooooooool!") various studies have employed some kind of normalization to alleviate the introduced lexical variety and input sparseness. We investigate the effects of lowercasing (LOW) and collapsing (COL) to ameliorate these problems. For collapsing we follow the following strategy: If the lowercased token is present in any of the MPQA, LIU or SWN sentiment dictionaries, we do no collapsing. If that is not the case, all adjacent duplicate characters are collapsed to two characters. If this collapsed version is in any of the lexical resources, it is kept, otherwise all duplicate characters are further collapsed into a token version with no adjacent duplicates. Table 7 shows the results of applying normalization.

	without	LOW	LOW+COL
MNB	65.96	66.01	66.16
SVM	69.57	70.26	**70.29**

Table 7: Effects of normalization on classifier performance (F_1).

We can observe that the normalization techniques result in a slight increase in classification performance. We expect this to play an even bigger role in later experiments, when we add the prior polarity of the lexical resources as features. In the following experiments we will use lowercasing and collapsing by default and come back to other versions of the token if needed.

Stemming

Stemming can be used to reduce morphological variety in the input. We use the Porter stemming algorithm (Porter, 1980), which strips common morphological suffixes from a given token. Table 8 shows the effects of stemming on classifier performance.
We can observe that also stemming increases classification performance slightly, which is why we use stems as main unigram features in the following exper-

	without	Stems
MNB	66.16	66.18
SVM	70.29	**70.96**

Table 8: Effects of stemming on classifier performance (F_1).

iments.

Negation handling

Handling negation is an important feature of sentiment analysis systems, as a negation can alter the sentiment of an expression significantly, in most cases reversing the sentiment.

We try two different methods of negation handling:

1. Attaching a negation marking to the words[27] directly before and after a negation word[28] (NEG1)

2. Attaching a negation marking to all words after a negation word until a punctuation sign or the end of the tweet (NEG2)

	without	NEG1	NEG2
MNB	66.18	66.99	65.57
SVM	70.96	**71.22**	71.19

Table 9: Effects of negation handling on classifier performance (F_1).

From the results in Table 9 we can see that negation handling indeed improves classification performance to some extent and strategy NEG1 works slightly better than approach NEG2.

[27] Words in this context mean tokens not containing punctuation

[28] *not, no, never, nobody, nothing, none, nowhere, neither*, words ending in *n't*

Twitter specific tokens

As many studies before have performed handling of Twitter specific tokens, we investigate the effect of inserting generic placeholders and splitting the hashtag symbol from the hashtag. We investigate the following three strategies:

- Replacing username mentions (tokens starting with "@") with a generic placeholder (USER)
- Replacing URLs (tokens starting with "http") with a generic placeholder (URL)
- Splitting the hashtag symbol from the rest of the tag ("#tag" → "#", "tag"), as this is not done by the tokenizer of Owoputi et al. (2013) (HASH)

	without	USER	URL	HASH	all
MNB	66.99	66.99	66.99	66.80	66.89
SVM	**71.22**	71.07	71.22	71.21	71.18

Table 10: Effects of Twitter specific token handling on classifier performance (F_1).

Interestingly, as we can see in Table 10, none of the three strategies of handling Twitter specific tokens actually improves the classification performance in our experiments. We nevertheless decide to keep the methods for the following experiments, as they don't hurt the performance significantly and might eliminate some bias in the training data.

Summary

We incrementally added different preprocessing methods to decrease lexical variety and handle negation in tweets. By doing this, the results on the ALL_3 dataset were increased about 1.74% F_1 for the Naive Bayes classifier and 3.75% F_1 for the support vector machine.

4.3 Features

In the feature experiments we will not incrementally add features, but evaluate the performance in comparison to the baseline for every feature separately. In the end we will present an ablation study on all features that were able to improve the performance over the baseline, to see how much they contribute to the final model. The baseline used is a classifier trained on unigrams with the preprocessing described in the previous section. Like the experiments on preprocessing, the feature experiments are performed on the ALL_3 dataset using 10-fold cross validation. In this section we concentrate on the SVM classifier, as it outperformed the MNB classifier in almost all previous experiments and can handle a mix of boolean, numeric and real-valued features well. In addition the learned weights for the features can give valuable information for debugging and enhancing the classifier's performance. This also gives us the space to provide a more detailed evaluation including the Precision (Prec) and Recall (Rec) values for the three classes.

N-grams

Several studies have tried to add n-gram features to improve classification performance, hoping to incorporate more context information than a plain bag-of-word model can hold. We add 2- to 4-grams to the unigram model.

	positive		neutral		negative		avg.
	Prec	Rec	Prec	Rec	Prec	Rec	F_1
without	**75.64**	62.24	69.89	**83.63**	73.02	65.61	71.18
+2	74.90	66.12	**71.81**	81.88	73.90	**67.64**	**72.46**
+2+3	74.52	**66.19**	71.48	81.92	**74.33**	66.75	72.26
+2+3+4	74.71	65.42	71.03	82.52	74.29	65.94	71.99

Table 11: Effects of adding n-gram features on classifier performance (SVM).

Table 11 shows the results, indicating that adding n-gram features does increase classification performance, with the model consisting of unigram plus bigram features performing best. Adding bigrams seems especially helpful

to enhance the recall of the positive and negative classes, and thus also to increase the precision of the neutral class. Looking at bigram features with high weights we find e.g. `i like` or `can't wait` as positive features for the positive class. The latter example addresses the reversal of sentiment due to negation. The former seems to disambiguate the verb like, which expresses clear positive sentiment, from other usage of the word like, which can be of type adjective, adverb, noun, conjunction, particle or interjection and does not carry strong positive sentiment. Another interesting bigram with high weight for the positive class is `look forward`, which combines two words without strong sentimental value into an expression indicating clearly positive anticipation of an event. This shows that n-grams can indeed provide valuable information for sentiment analysis, especially in the case of tweets, where due to the length restriction a bigram like the last example might be the one and only expression of sentiment in the whole message. Using n-gram features without using the corresponding lower order n-grams (e.g. bigrams, but not unigrams) resulted in clearly lower classification performance.

Sentiment Lexica

To incorporate information from sentiment lexica in the classifier, we compute a prior polarity of the tweet by counting all positive and negative words present in the tweet and in a sentiment lexicon, and assign the tweet a prior polarity if one count is greater than the other. We use the MPQA subjectivity lexicon (MPQA), Bing Liu's opinion lexicon (LIU), SentiWordNet (SWN), and the Twitrratr wordlist (TRR). For further information about these resources see Section 3.2.

From the results in Table 12 we can see, that all lexica can provide an increase in classification performance, with Bing Liu's opinion lexicon (LIU) yielding the highest improvement. A quite surprising result is that the Twitrratr wordlist (TRR) improves the result more than SentiWordNet or the MPQA subjectivity lexicon, despite being significantly smaller. This can be seen as evidence of the need for more Twitter specific sentiment lexica, as the lexicon yielding the highest improvement also contains common misspellings

	positive		neutral		negative		avg.
	Prec	Rec	Prec	Rec	Prec	Rec	F_1
without	75.64	62.24	69.89	83.63	73.02	65.61	71.18
MPQA	75.28	63.23	70.26	82.78	73.11	66.32	71.43
LIU	75.05	64.81	71.40	81.90	73.39	68.19	72.18
SWN	75.48	62.45	69.89	83.41	73.49	65.96	71.31
TRR	75.69	63.47	70.54	82.89	72.77	66.54	71.59
all	75.52	65.35	71.77	81.83	73.48	68.97	**72.56**

Table 12: Effects of adding sentiment lexicon features on classifier performance (SVM).

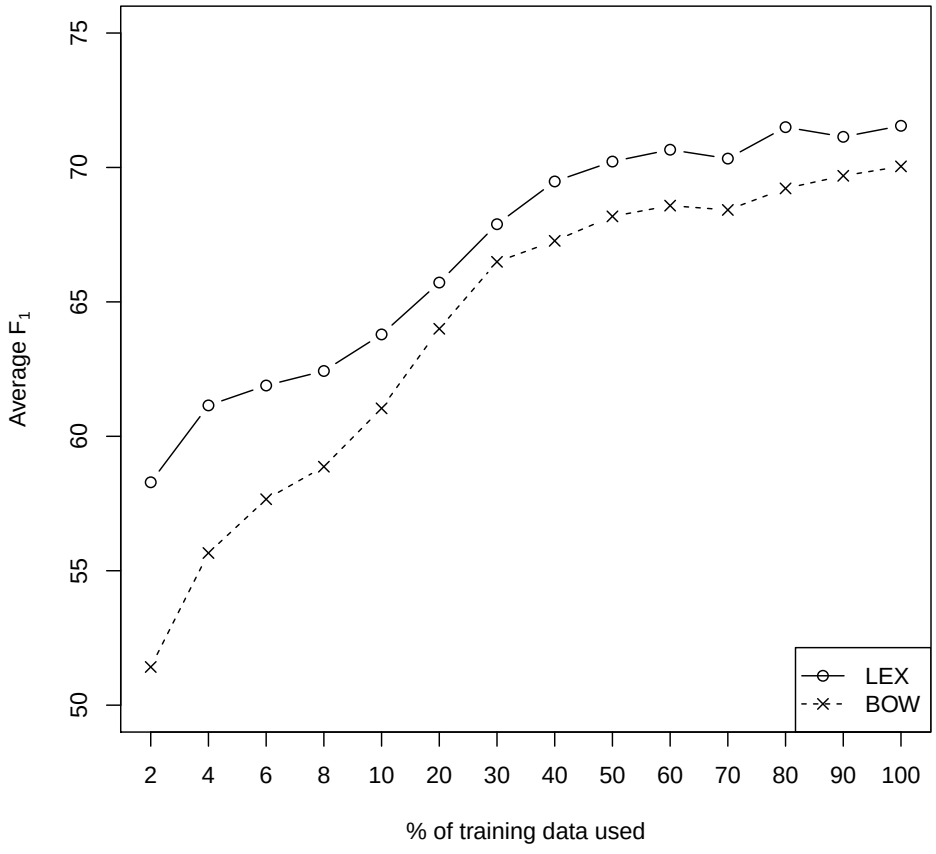

Figure 1: Effect of adding lexicon features on different sized training sets

and other phrases used in social media.

Figure 1 shows the impact of using sentiment lexicons (LEX) on different sized portions of the training set. Not surprisingly, the gain in classification performance compared to the baseline (BOW) is much larger when less training data is available.

In Section 4.2 we found lexical recall to be slightly higher when using the RE1 tokenization strategy. Computing the prior polarities of the tweet on the collapsed token extracted with this strategy improves the average F_1 to 72.61%.

Part-of-speech tags

While some previous studies reported improved results when using part-of-speech tags, others did not find such effects. This might have been the result of using common part-of-speech taggers, which, being trained on other data, might have had problems with Twitter specific tokens and grammar resulting in inaccurate part-of-speech tags. To address this problem, we use the part-of-speech tagger by Owoputi et al. (2013), which comes with a tagset tailored for Twitter peculiarities and is trained on Twitter data. We experiment with adding a boolean presence/absence feature for each tag to the features (POSFEAT), marking every unigram stem with its part-of-speech tag (POSSTEM) and adding those part-of-speech tag marked unigram stems as additional features to the normal unigram stems, instead of replacing them (+POSSTEM).

From the results given in Table 13 we can see that part-of-speech tags - at least when using an appropriate part-of-speech tagger - can indeed improve classification performance. This results from a boost of recall of the positive and negative classes and an improvement of precision for the neutral class. A marginal improvement can be achieved by providing information about the presence of certain part-of-speech tags in the tweet (POSFEAT). While replacing normal unigram features with their part-of-speech tag augmented counterparts (POSSTEM) lowers classification performance, adding them as additional features (+POSSTEM) proves to be a valuable source of

	positive		neutral		negative		avg.
	Prec	Rec	Prec	Rec	Prec	Rec	F_1
without	**75.64**	62.24	69.89	83.63	73.02	65.61	71.18
POSFEAT	75.37	62.26	70.16	83.34	73.05	66.43	71.32
POSSTEM	75.64	61.71	69.49	**83.80**	73.05	65.14	70.94
+POSSTEM	74.74	64.98	71.54	81.76	**73.19**	67.96	72.10
all	74.63	**65.11**	**71.71**	81.35	72.97	**68.52**	**72.15**

Table 13: Effects of adding part-of-speech tag features on classifier performance (SVM).

information. Examining features with high weights, we can observe that our earlier intuition about n-grams also resolving part-of-speech ambiguity seems to be right, as `V=like` is a strong feature for the positive class, while other occurences of the word "like" are not. The best result in the experiments is achieved when adding both POSFEAT and +POSSTEM features (all).

Occurence of Emphasis

In informal texts, such as tweets, authors sometimes use emphatic uppercasing or lengthening to express especially strong sentiment. We try different ways of capturing these phenomena. We compute the ratio of uppercase letters to lowercase letters in the tweet (UPRATIO), and add a feature indicating the occurrence of more than three adjacent repetitions of the same character in the tweet (REPEMPH). Furthermore, we combine part-of-speech tags with emphasis information, by adding features that represent the presence of a part-of-speech tag with the corresponding token being either all uppercase (UPPOS) or carrying a character repetition of more than three adjacent letters (REPPOS).

Looking at the results given in Table 14 we can see that features capturing emphasis information can result in a minor increase of classification performance. However, there are no features with very high feature weights. Additionally, when adding all emphasis features at the same time the overall performance is not higher than before. This indicates that emphasis infor-

	positive		neutral		negative		avg.
	Prec	Rec	Prec	Rec	Prec	Rec	F_1
without	75.64	62.24	69.89	83.63	73.02	65.61	71.18
UPRATIO	75.69	62.24	69.93	83.60	72.99	65.78	71.22
REPEMPH	75.67	62.24	69.91	83.60	73.05	65.76	71.22
UPPOS	75.55	62.40	70.05	83.66	73.01	65.59	71.23
REPPOS	75.69	62.30	69.90	83.62	73.13	65.72	71.24
all	75.48	62.34	69.98	83.51	72.94	65.64	71.18

Table 14: Effects of adding emphasis features on classifier performance (SVM).

mation is not a crucial carrier of sentiment information, but is rather used to strengthen a sentiment expressed by verbal means.

Word Clusters

Word clusters built by Brown clustering (Brown et al., 1992) have been used to enhance performance of several natural language processing tools, such as part-of-speech taggers (Owoputi et al., 2013), chunking and named entity recognition systems (Turian et al., 2010). However, their use in sentiment analysis for Twitter is new. We learn weights for the clusters of Owoputi et al. (2013), who performed Brown clustering on over 56 million tweets (+WC), by checking the presence of the raw, normalized and collapsed token in the clusters. As we can see in Table 15, adding word cluster features is very useful to improve the recall of positive and negative classes, resulting in an overall increase of classification performance. Looking at some clusters,[29] we can see that different spellings of the same word tend to be grouped together in the same cluster, as well as semantically related words and even emoticons.[30] This makes word clusters an especially useful tool for sentiment analysis in Twitter, as they address the problem of spelling variety. Furthermore, as they are produced from unlabeled data, they can be

[29]see `http://www.ark.cs.cmu.edu/TweetNLP/cluster_viewer.html`
[30]see Appendix B for some illustrative examples.

recomputed easily and thus be used to address new concepts in the Twitter stream, which might not be present in the training data.

	positive		neutral		negative		avg.
	Prec	Rec	Prec	Rec	Prec	Rec	F_1
without	**75.64**	62.24	69.89	**83.63**	**73.02**	65.61	71.18
+WC	75.53	**64.98**	**71.53**	82.21	72.83	**67.74**	**72.18**

Table 15: Effects of adding word cluster features on classifier performance (SVM).

Ablation study

As the information covered by certain features sometimes overlaps (e.g. in the previous example the ambiguity of the word "like", which can be addressed by n-grams as well as pos-tag features) we conduct an ablation study. This tests which of the previously separately tested feature groups are most helpful to a model combining all of them.

As can be seen in Table 16, adding all previously tested features results in an increase of overall classification performance of about +2.34% average F_1. Removing one group of features at a time to see their contribution to the final model shows that bigrams, sentiment lexicon features and word clusters contribute to the final model, as the score drops when removing those groups. However, removing the part-of-speech and emphasis features results in an increase of overall classification performance, despite the fact that those features did improve performance in previous experiments when adding them to the bag-of-words model. While part-of-speech tag features seem not to be helpful for the final model, removing them from a model without n-grams lowers classification performance slightly, indicating that they carry information that is missing when not considering n-gram features, as previously suspected. The best performance was reached when using stem unigrams with bigram, sentiment lexicon and word cluster features, disregarding emphasis and part-of-speech tag features.

	positive		neutral		negative		avg.
	Prec	Rec	Prec	Rec	Prec	Rec	F_1
without	75.64	62.24	69.89	83.63	73.02	65.61	71.18
all	74.10	68.92	73.93	79.64	73.94	71.03	73.52
-2GRAM	74.44	67.86	73.38	79.82	73.39	70.97	73.21
-LEX	74.01	67.90	73.16	80.12	73.29	69.38	72.86
-POS	74.79	68.84	73.98	80.33	74.22	71.16	73.79
-EMPH	74.27	69.16	74.02	79.63	73.99	71.16	73.63
-WC	74.14	68.78	73.44	79.91	74.15	70.07	73.32
-2GRAM -POS	75.32	66.41	72.75	81.16	73.19	70.22	73.00
-EMPH -POS	74.87	68.98	74.03	80.37	74.24	71.12	**73.84**

Table 16: Ablation study (SVM).

Summary

In this section we conducted experiments testing the usefulness of different features for sentiment analysis in Twitter. When being added to a bag-of-words model separately, all tested feature groups were able to improve the model's performance. However, an ablation study revealed that in a combined model bigrams, sentiment lexicon and word cluster features were helpful to the model, while part-of-speech tag features as well as features covering emphatic lengthening and uppercasing were not. The performance of the best model was +2.66% F_1 higher than the baseline without additional features.

4.4 Methods

Two step classification

Some previous studies have addressed the 3-class classification problem of positive vs. neutral vs. negative sentiment in two steps: First, classifying a tweet as subjective or objective (which is in previous work then used as the neutral class) and subsequently classifying all subjective tweets as either positive or negative. We test whether this strategy is superior to directly classifying a tweet as one of the three classes.

	1 step		2 steps	
	Acc	avg. F_1	Acc	avg. F_1
MNB	69.92	69.64	67.85	67.95
SVM	**74.34**	**73.84**	73.37	72.92

Table 17: Performance of classifying in one or two steps.

As we can see from Table 17, neither the Naive Bayes classifier nor the support vector machine benefits from the two step strategy in our experiment.

Training Set Size

In this experiment we tested the impact of training set size on classifier performance. We took a random sample of 20% (4351 tweets) from the ALL_3 dataset and used it as a test set. Then we trained different classifiers on different sized portions of the remaining 80% (17,403 tweets). For this experiment we added another classification algorithm: the perceptron. While the perceptron is a linear classifier like the support vector machine and is in fact very similar to the implementation we used, it does not make use of more advanced machine learning techniques such as regularization or a changing learning rate. Also, while the support vector machine uses a large margin loss function (hinge loss), the perceptron uses a zero-one loss function, which updates the weights of the classifier only on misclassified examples during the training process. In the experiment, we compare the support vector machine

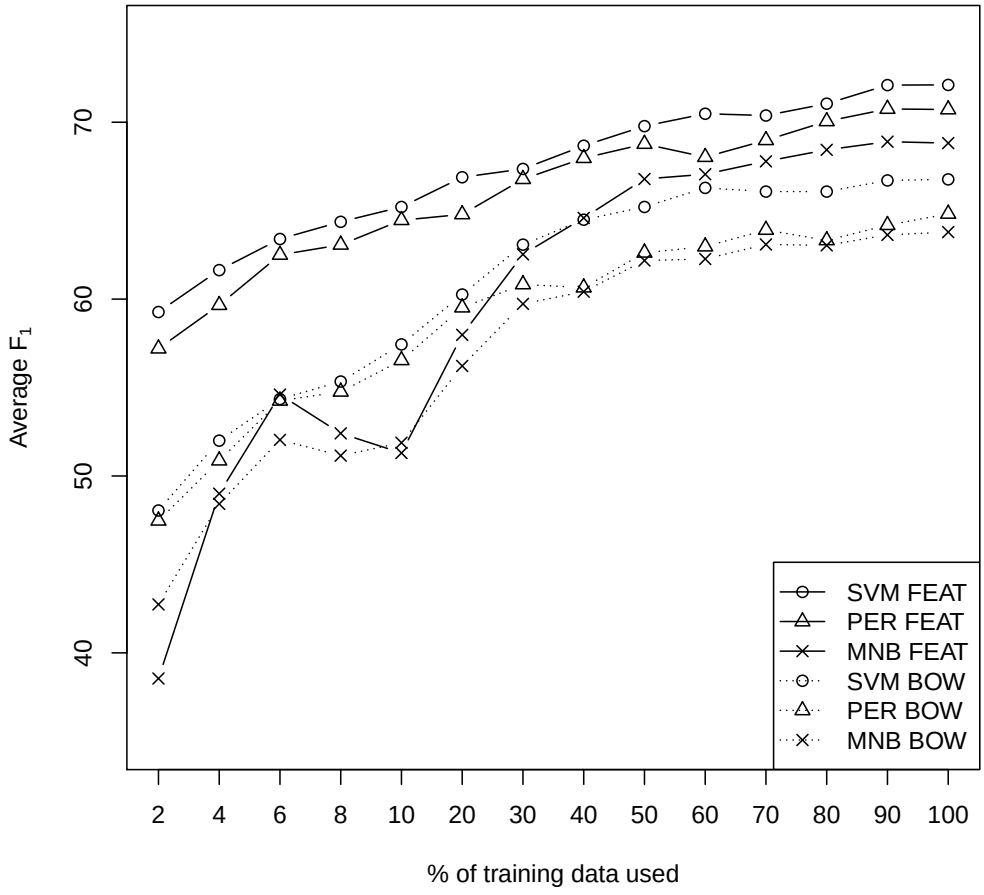

Figure 2: Impact of training set size on classifier performance

(SVM), the perceptron (PERC) and the Naive Bayes (MNB) classifier, each trained with the full feature set (FEAT) as well as only on unigrams (BOW). The results can be seen in Figure 2.

We can make several interesting observations in the results. As expected, the models using the full feature set do - with the exception of the Naive Bayes classifier - outperform the bag of word unigram models. Concerning the tested algorithms, we can see that the support vector machine outperforms the respective other classifiers in all cases, followed by the perceptron algorithm and the Naive Bayes classifier performing worst. The larger gap between the classifiers trained with and without the full feature set on the left side of the graph indicates that the extra features used are especially useful when little training data is available. However, this seems not to be true for the Naive Bayes classifier. Furthermore, we can observe that while with the full feature set the performance of perceptron and support vector machine is quite similar, the latter can benefit more from more training data in the case of the bag of words model, widening the performance gap when more than 30% of the training data is used.

Semi-supervised learning

Sometimes it is not possible to create large manually labeled datasets for training, e.g. in case of limited data available or limited resources to gather and annotate a lot of data. In the case of Twitter, getting data is not the problem, as a huge amount of tweets can be downloaded through the Twitter API. However, creating manually labeled training data is expensive. In semi-supervised learning, the available manually labeled data is combined with unlabeled data, to improve classifier performance over the purely supervised classifier. We tested self-learning (Nigam and Ghani, 2000) and tri-learning (Zhou and Li, 2005), with an additional dataset of over 100,000 unlabeled tweets. The basic idea of both algorithms is to first train supervised classifier(s) on the labeled data, then using the classifier(s) to classify the unlabeled data and retrain the classifier(s) on the combined data. While this is a very interesting approach, we did not find any significant improvement in

performance, despite trying various combinations of settings, classifiers and variants of the algorithms.

Random Subspace Learning

Training on random subspaces of the training data has been used before to improve the robustness of classifiers in natural language processing applications (Søgaard and Johannsen, 2012). However, there are no publications on using random subspace learning for sentiment analysis of tweets. The intuition behind this method is to randomly disable features for each training sample, which leads to more redundancy in the models by forcing the classifier to not rely too much on very strong features. We try random subspace learning, by training classifiers on 10 concatenated corrupted copies of the ALL_3 dataset, where in each copy each feature of each sample is disabled with a probability of p. Figure 3 shows the result for the Naive Bayes classifier and support vector machine for different values of p. From the results we can see that both classifiers marginally benefit from applying random subspace learning. However, while the Naive Bayes classifier reaches its highest performance with a very corrupted dataset (when each feature is disable with a probability of 80%), the support vector machine reaches its best result when each feature is deactivated with a probability of only 20%.

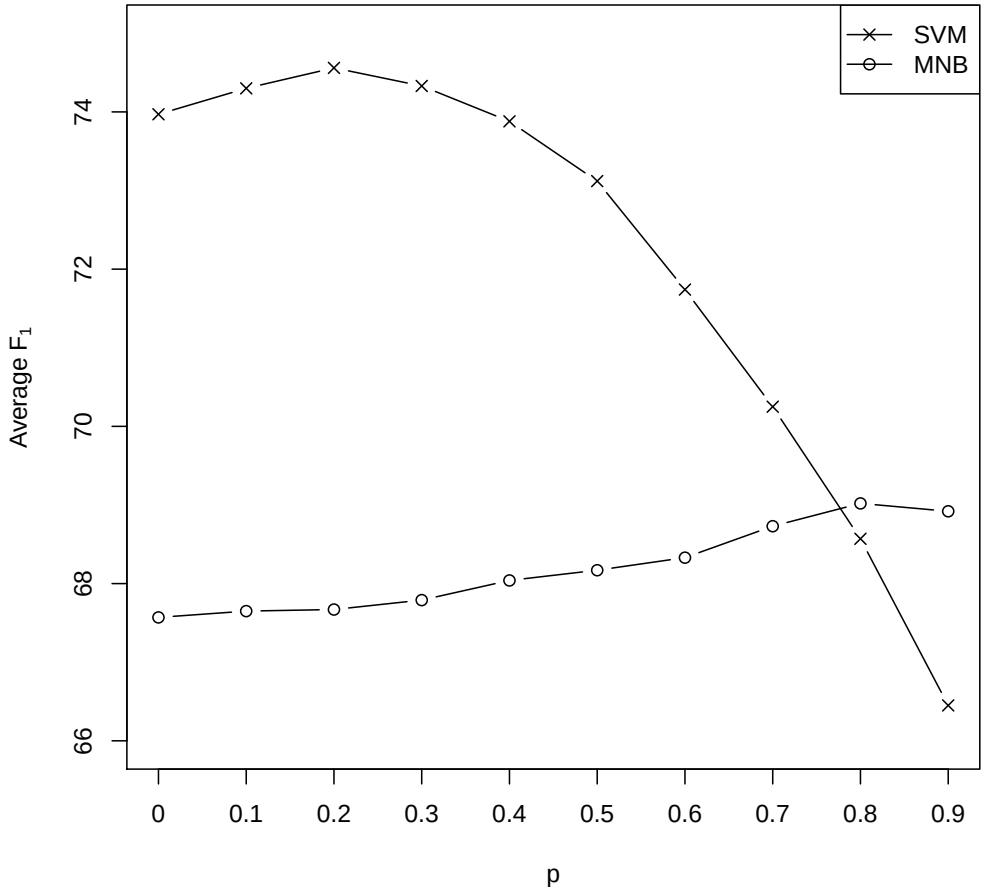

Figure 3: Effect of random subspace learning on classification performance

4.5 Final Model Evaluation

For the final model, we choose the support vector machine classifier, as it outperformed the other tested methods in all our experiments. We combine all techniques which increased classifier performance in the previous experiments. To summarize, the preprocessing for the final model consists of tokenizing the tweets using the Twitter specific tokenizer of Owoputi et al. (2013). The tokens are normalized by lowercasing and collapsing of adjacent repeated characters, if the uncollapsed token is not found in one of the used lexical resources. The tokens are stemmed using the Porter stemming algorithm and a negation marker is attached to the token before and after a negation word. As features these tokens are used as unigram and bigram features. Furthermore, four different sentiment lexica are used to compute prior polarities of the tweet, which are used as features for the classifier. We also register the presence or absence of the tokens in word clusters computed on Twitter data as additional features. Finally, we use random subspace learning (p = 0.2) to improve the robustness of our final model. Table 18 shows the results of the final model in comparison to the bag of words baseline.

	Baseline		Final Model	
Dataset	Acc	F1	Acc	F1
HCR_2	76.17	68.97	79.55	72.43
OMD_2	80.57	78.67	85.77	84.58
SAN_2	78.80	78.74	85.58	85.55
SEM_2	81.38	75.83	87.75	84.66
ALL_2	80.30	79.76	86.24	**85.91**
HCR_3	65.76	58.50	69.31	61.30
SAN_3	77.33	65.26	80.68	72.17
SEM_3	67.46	61.84	73.76	71.00
ALL_3	68.87	67.47	74.90	**74.56**

Table 18: Evaluation of final model on different datasets

We also test our final model, trained on the ALL_2 dataset, with the STS_{test} dataset. The results are 86.07% accuracy and 86.02% average F1, being the best reported result on this dataset so far.

Most informative features

Analyzing the features with the highest weights in the final model, the results are quite intuitive.

For the **positive** class, highly weighted unigram stems include `excit`, `enjoy`, `thank`, `happi` and `love`. Bigrams include `i like`, `can't wait`, `the best`, `look forward` and `don't miss`. Regarding word clusters, the highest ranked clusters contain words such as `awesome`, `priceless`, `happy`, `excited`, `cool` and `nice`. Furthermore, emoticons like `:)`, `:D` and `<3` are included in the features with very high weights, as well as prior positive polarity from the sentiment lexica. All these expressions indicate very positive sentiment, so it is not surprising and serves as sanity check for the classifier that these features have strong positive weight towards their classes. Also the features with high negative weights for each class make sense, as they usually are strong positive features for one of the other two classes. This is also true for the neutral and negative classes.

For the **neutral** class, highly weighted unigram stems include `twitter`, `wear`, `announc` and `latest`. Bigrams include `on twitter`, `it may`, `night twitter`, `this saturday`, `do you`. Highly ranked word clusters include words such as `twitter`, `understand`, `someone`, `thinking` and `stay`. It is also notable that emoticons and prior sentiment lexicon polarity features are only present as high negative weights. As in the positive class, these results make sense intuitively. From the highly ranked features we can infer that neutral tweets often seem to be about Twitter and the users' communication on the platform itself, status updates of private persons writing about what they are wearing or making plans e.g. for the weekend, accounts of news sites that report on company announcements for instance and conversations about or advertisements for products.

For the **negative** class, highly weighted unigram stems include `fuck`, `suck`, `sad`, `fail` and `hate`. Bigrams include `i hate`, `may never`, `the hell` and `the worst`. Word clusters with high positive weights include words such as `sad`, `frustrating`, `stupid`, `dumb`, `ignorant` and `rude`. The highest rated feature is the emoticon `:(`, and prior negative polarity from sentiment lexica

is a strong indicator for the negativity of the tweet as well.

While all these features are fairly general and should give good indication for sentiment regarding a wide range of subjects, there are also highly ranked features specific to the training data, especially the political OMD and HCR datasets. Tokens like `+1` and `-1` seem to be commonly used to express agreement or disagreement with one of the candidates positions in the OMD dataset, and there are also very specific terms present, such as the hashtag `#killthebill`, referring to the health care reform debate.

Error analysis

Looking at the confusion matrix (Figure 4), we can see that most tweets are confused between the neutral class and one of the subjective classes. However, sometimes it might be hard for the annotator to decide whether a message contains sentiment or not, and different annotators can have different opinions about this matter. The distinction between positive and negative sentiment is often quite clear and probably the most undesired error of a sentiment analysis system, so we analyzed some mistakes that the classifier made between those two classes.

```
          |    -           |
          |    1    0    1 |
        ---+----------------+
        -1 |<3959> 834  557 |
         0 |  800<7155>1274 |
         1 |  657 1339<5179>|
        ---+----------------+
    (row = reference; col = test)
```

Figure 4: Confusion matrix of 10-fold cross validation on ALL_3 dataset.

One problem seems to be the detection of the strongest overall sentiment, when positive and negative indicators are present in the tweet, such as in "*I like Trey Burke but I think pre-season 1st team all-american is a bit over the top.*", which was classified as positive, but annotated as negative in the gold

standard. As it is not always clear which is the stronger sentiment, document-level sentiment analysis is probably too imprecise to be able to tackle this kind of message. Another example of this problem is the tweet "@*erickmartin oh wow! Did you just trade Elvis? I'm just saying @the_chance44 may not like you! :)*", which was also classified as positive, but annotated as negative. In this example, the emoticon `:)` indicates strong positive sentiment, while the phrase "*may not like*" expresses the opposite. Whether this tweet was actually meant more positively or negatively can probably only be determined with knowledge of the context of the conversation between the users. Another problem in this context are quotes, such as in the tweet "*How are these fools talking about "Niners are the best team in the NFC"*", in which the author opposes the opinion of somebody else by calling them fools.

Another difficulty seem to be tweets that contain irony or sarcasm as in "*New outfit to wear $500... Being a presidential candidate and not knowing how to tie a tie.... PRICELESS! #current*" or expressing schadenfreude as in "*#tweetdebate McPain is getting pwnd ahahah*", where a negative event (someone "getting owned") is followed by an evil laugh expression, indicating the approval of the author. This example could be addressed by separate analysis of the opinion of the author towards the mentioned person and the event, when moving from document-level sentiment analysis to entity-/aspect-level sentiment analysis. The negative sentiment is strengthened by a deliberate misspelling of the name McCain, changing it to McPain.

Also, more sophisticated negation handling strategies might be needed to correctly classify sentiment in all tweets. While the strategy used (marking words with a negation marker if they occur directly next to a negation) yielded better overall results, it is not sufficient to correctly classify every tweet, such as "*Brilliant 85 mins for @naisy14 in the Mersey derby. Not a bad game for 1st goal....*", which was classified as negative, but in reality expresses positive sentiment.

5 Discussion and Conclusions

In the previous section, we reported results on old and new techniques for sentiment analysis of Twitter messages (tweets). The experiments were executed as a 10-fold cross-validation on the largest manually annotated dataset used for this problem so far, which was obtained by combining several datasets from previous studies. These characteristics make the findings a valuable source of information for researchers and engineers trying to build sentiment analysis systems, by giving possible directions regarding techniques that were not compared before, not used before at all, or for which conflicting results have been reported in previous studies. We conducted experiments on preprocessing methods, feature extraction and machine learning methods.

Concerning machine learning techniques, we find a support vector machine classifier to outperform other tested classifiers. Based on the results of reported and not reported experiments, we recommend using linear classifiers with large-margin loss functions and advanced techniques such as regularization and an adaptive learning rate, such as a support vector machine or logistic regression, for the task of sentiment analysis of microblogs. Their performance can be improved by employing random subspace learning and supplying more training data.

While an interesting approach, we could not find semi-supervised learning techniques to improve classification performance using self-training and tri-training. However, word clusters learned in an unsupervised manner proved to be useful features, which can be seen as some kind of semi-supervised method. It should be determined in more detailed future research whether these or other semi-supervised methods might not in fact be useful for the task of sentiment analysis in Twitter. The fact that large amounts of unlabeled data are available seems very promising to save the costs of creating large manually annotated training sets. To address this problem, domain adaption strategies should also be considered, employing existing out-of-domain datasets.

Furthermore, we could not find any improvement by conducting the 3-class classification task in 2 steps, which was previously attempted by Bar-

bosa and Feng (2010), Jiang et al. (2011), Wang et al. (2011) and Liu et al. (2012), but rather a decrease in performance when using this approach. While it might make intuitive sense to address the 3-class problem as the two separate steps of subjectivity and polarity classification, it does not make sense from a machine learning point of view, when the desired outcome is a classification with 3 classes. While for a very high number of target classes the classification in steps might be preferable, 3 classes are usually manageable directly. We do not believe that there are a lot of features that would be useful to one of the steps but harmful to the other step, which would be the only reason we could think of for separating the steps.

Regarding tokenization we find the Twitter-specific tokenizer by Owoputi et al. (2013) to work better than more simple approaches, such as splitting on whitespace or using single regular expressions. This confirms that the effort made by the authors had a good cause and the use of their tokenizer can be recommended when dealing with Twitter data. However, when looking up tokens in a sentiment lexicon, higher recall can be achieved by only considering alphabetic characters in the tokens.

Because of the informal nature of tweets, spelling of the words can vary a lot, which is why we find normalization to be an important step. To deal with this lexical diversity we employ lowercasing and collapsing of repeated adjacent letters, when a token is not present in one of the lexical resources used. We find that stemming, reducing lexical variety further on a morphological level, has a positive effect of classification performance.

When comparing simple negation handling strategies, we find attaching a marker to tokens directly before or after a negation word to yield the best overall results. However, the analysis of misclassified tweets of the final model revealed that more sophisticated negation handling strategies are needed to identify the sentiment of all tweets correctly. Further research could include negation scope detection in Twitter messages, using part-of-speech tags and dependency relations.

While it is common to replace Twitter-specific tokens such as usernames and URLs with generic placeholders, we did not find any actual improvement of classification performance by doing so. However, there was no significant

decrease in performance either, so this technique can be used to reduce the feature space of the training data, which might improve speed and memory usage of machine learning classifiers.

Altogether, the improvements made by applying the mentioned preprocessing techniques resulted in a greater improvement of classification performance than feature engineering or applying additional learning techniques, which stresses the importance of proper preprocessing of the noisy and diverse Twitter data.

When testing features, we find adding bigrams to improve classification performance over just using unigrams for the classifiers. Observing highly weighted bigrams we were able to see that they address part-of-speech ambiguities and cover sentiment expressions that are specific to phrases, not words.

While we find that features covering part-of-speech tags were not useful for our final model, we did find that they can improve classification performance of a bag-of-words model and when no n-grams are used. So the usefulness of part-of-speech tags for this task seems to be dependent on the other information sources supplied to the classifier. This might explain the contradicting results reported by Go et al. (2009) and Kouloumpis et al. (2011), who do not find improvement by using them, and Pak and Paroubek (2010), Barbosa and Feng (2010), Bermingham and Smeaton (2010) and Agarwal et al. (2011), who find at least small improvements by using part-of-speech tags in their features, or don't report the contrary.

The results of our experiments on using features covering emphasis expressed by uppercasing or word lengthening indicate that these techniques are used to strengthen sentiment rather than express sentiment, which is why we did not include them in our final model.

Using prior polarity features computed using four different sentiment lexica did prove to be a valuable source of information. The creation of such resources is highly desirable, if not already present for the language, or if needed, a specific target domain.

A new source of information for the task at hand are word clusters obtained by Brown clustering of large amount of tweets. These clusters proved

to be highly useful for enhancing the recall of sentiment expressions in Twitter data, as they not only group semantically related words together, but are also useful to cover the wide variety in spelling, as differently spelled versions of the same word often end up in the same cluster. As they can be computed in an unsupervised manner we highly recommend using Brown word clusters for sentiment analysis in Twitter. They are a cheap way to address lexical brittleness and the incorporation of new words, which might not be present in annotated training data. Using these word clusters was probably one of the reasons for the good results obtained by our system submitted to the SemEval-2013 shared task (Günther and Furrer, 2013, in press; Nakov et al., 2013, in press). For more information about using Brown word clusters see Owoputi et al. (2013), Turian et al. (2010) and Brown et al. (1992). It would also be interesting to see if sentiment analysis systems using a purely lexical approach can benefit from using these word clusters, using them to assign prior polarity values to spelling variants not present in the sentiment lexicon itself.

When analyzing our final model, which combines all helpful strategies identified in the previous experiments, we find the features learned by the classifier to make sense intuitively, which is a good sign regarding the success of the intended learning task. Outperforming the baseline by more than 7 F_1 %-points can be seen as a considerable achievement, as beating a bag-of-words baseline in document classification problems is often considered a rather difficult task (Moschitti and Basili, 2004). The performance of the final model on datasets used by previous work in the field is higher than the originally reported results in almost all cases (compare Table 1, 2 and 18), which is proof of the effectiveness of the proposed method.

We believe that the most promising direction of future research in the field of sentiment analysis of microblogs lies in more fine-grained approaches, such as sentiment analysis on entity and aspect level. Twitter specific natural language processing tools such as dependency parsers and negation scope detectors will be needed to achieve the best results for these tasks and strategies for identification of sentiment targets in tweets will have to be explored. The lack of general domain datasets annotated for sentiment analysis on en-

tity/aspect level might be the biggest hurdle to overcome. The research of semi-supervised learning and domain adaption techniques for this task is also very desirable, as the large amounts of easily available unlabeled data provide a perfect setting for these methods. Finally, the creation of Twitter-specific sentiment lexica, possibly by combining available word lists and spelling variants identified by techniques such as Brown clustering, could be of great value for sentiment analysis of microblog data.

References

Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. Sentiment analysis of twitter data. In *Proceedings of the Workshop on Languages in Social Media*, pages 30–38. Association for Computational Linguistics, 2011.

Fotis Aisopos, George Papadakis, and Theodora Varvarigou. Sentiment analysis of social media content using n-gram graphs. In *Proceedings of the 3rd ACM SIGMM international workshop on Social media*, pages 9–14. ACM, 2011.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the 7th conference on International Language Resources and Evaluation (LREC'10), Valletta, Malta*, 2010.

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

Adam Bermingham and Alan F Smeaton. Classifying sentiment in microblogs: is brevity an advantage? In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1833–1836. ACM, 2010.

Albert Bifet and Eibe Frank. Sentiment knowledge discovery in twitter streaming data. In *Discovery Science*, pages 1–15. Springer, 2010.

Samuel Brody and Nicholas Diakopoulos. Cooooooooooooooollllllllllll!!!!!!!!!!!!!!: using word lengthening to detect sentiment in microblogs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 562–570. Association for Computational Linguistics, 2011.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479, 1992.

Jacob Cohen et al. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

Anqi Cui, Min Zhang, Yiqun Liu, and Shaoping Ma. Emotion tokens: Bridging the gap among multilingual twitter sentiment analysis. In *Information Retrieval Technology*, pages 238–249. Springer, 2011.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 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, 2010.

Christiane Fellbaum. Wordnet: An electronic database, 1998.

Joseph L Fleiss, Bruce Levin, and Myunghee Cho Paik. The measurement of interrater agreement. *Statistical methods for rates and proportions*, 2: 212–236, 1981.

Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 2009.

Tobias Günther and Lenz Furrer. Gu-mlt-lt: Sentiment analysis of short messages using linguistic features and stochastic gradient descent. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 328–332, Atlanta, Georgia, USA, June 2013, in press. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S13-2054`.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. Target-dependent twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 151–160, 2011.

Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. Twitter sentiment analysis: The good the bad and the omg. In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*, pages 538–541, 2011.

Akshi Kumar and Teeja Mary Sebastian. Sentiment analysis on twitter. *IJCSI International Journal of Computer Science Issues*, 9(3):372–378, 2012.

Chenghua Lin and Yulan He. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 375–384. ACM, 2009.

Bing Liu. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167, 2012.

Kun-Lin Liu, Wu-Jun Li, and Minyi Guo. Emoticon smoothed language models for twitter sentiment analysis. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

Micol Marchetti-Bowick and Nathanael Chambers. Learning for microblogs with distant supervision: political forecasting with twitter. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 603–612. Association for Computational Linguistics, 2012.

Shotaro Matsumoto, Hiroya Takamura, and Manabu Okumura. Sentiment classification using word sub-sequences and dependency sub-trees. In *Advances in Knowledge Discovery and Data Mining*, pages 301–311. Springer, 2005.

Diana Maynard, Kalina Bontcheva, and Dominic Rout. Challenges in developing opinion mining tools for social media. *Proceedings of NLP@ can u tag# user_generated_content*, 2012.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 523–530. Association for Computational Linguistics, 2005.

George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

Akshay Minocha and Navjyoti Singh. Generating domain specific sentiment lexicons using the web directory. *Advanced Computing: an International Journal*, 3, 2012.

Alessandro Moschitti. Efficient convolution kernels for dependency and constituent syntactic trees. In *Machine Learning: ECML 2006*, pages 318–329. Springer, 2006.

Alessandro Moschitti and Roberto Basili. Complex linguistic features for text classification: A comprehensive study. In *Advances in Information Retrieval*, pages 181–196. Springer, 2004.

Preslav Nakov, Sara Rosenthal, Zornitsa Kozareva, Veselin Stoyanov, Alan Ritter, and Theresa Wilson. 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, June 2013, in press. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S13-2052`.

Finn Årup Nielsen. A new ANEW: Evaluation of a word list for sentiment analysis in microblogs. In *Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages*, volume 718 of *CEUR Workshop Proceedings*, pages 93–98, 2011.

Kamal Nigam and Rayid Ghani. Analyzing the effectiveness and applicability of co-training. In *Proceedings of the ninth international conference on Information and knowledge management*, pages 86–93. ACM, 2000.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of NAACL 2013*, 2013.

Brendan O'Connor, Ramnath Balasubramanyan, Bryan R Routledge, and Noah A Smith. 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, 2010.

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

Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135, 2008.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 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, 2002.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Sasa Petrovic, Miles Osborne, and Victor Lavrenko. The edinburgh twitter corpus. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in a World of Social Media*, pages 25–26, 2010.

Martin F Porter. An algorithm for suffix stripping. *Program: electronic library and information systems*, 14(3):130–137, 1980.

Alan Ritter, Sam Clark, Oren Etzioni, et al. Named entity recognition in tweets: an experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534. Association for Computational Linguistics, 2011.

Hassan Saif, Yulan He, and Harith Alani. Semantic smoothing for twitter sentiment analysis. In *Proceeding of the 10th International Semantic Web Conference (ISWC)*, 2011.

Hassan Saif, Yulan He, and Harith Alani. Alleviating data sparsity for twitter sentiment analysis. In *Proceedings of the 2nd Workshop on Making Sense of Microposts*, 2012a.

Hassan Saif, Yulan He, and Harith Alani. Semantic sentiment analysis of twitter. In *The Semantic Web–ISWC 2012*, pages 508–524. Springer, 2012b.

David A Shamma, Lyndon Kennedy, and Elizabeth F Churchill. Tweet the debates: understanding community annotation of uncollected sources. In *Proceedings of the first SIGMM workshop on Social media*, pages 3–10. ACM, 2009.

Anders Søgaard and Anders Johannsen. Robust learning in random subspaces: Equipping nlp for oov effects. In *COLING (Posters)*, pages 1171–1180, 2012.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. A machine learning approach to coreference resolution of noun phrases. *Computational linguistics*, 27(4):521–544, 2001.

Michael Speriosu, Nikita Sudan, Sid Upadhyay, and Jason Baldridge. Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proceedings of the First Workshop on Unsupervised Learning in NLP*, pages 53–63. Association for Computational Linguistics, 2011.

Philip J Stone, Dexter C Dunphy, and Marshall S Smith. The general inquirer: A computer approach to content analysis. 1966.

Mike Thelwall, Kevan Buckley, Georgios Paltoglou, Di Cai, and Arvid Kappas. Sentiment strength detection in short informal text. *Journal of the American Society for Information Science and Technology*, 61(12):2544–2558, 2010.

Mike Thelwall, Kevan Buckley, and Georgios Paltoglou. Sentiment strength detection for the social web. *Journal of the American Society for Information Science and Technology*, 63(1):163–173, 2012.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

Xiaolong Wang, Furu Wei, Xiaohua Liu, Ming Zhou, and Ming Zhang. Topic sentiment analysis in twitter: a graph-based hashtag sentiment classification approach. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1031–1040. ACM, 2011.

Cynthia Whissell. The dictionary of affect in language. *Emotion: Theory, research, and experience*, 4(113-131):94, 1989.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210, 2005.

Ley Zhang, Riddhiman Ghosh, Mohamed Dekhil, Meichun Hsu, and Bing Liu. Combining lexiconbased and learning-based methods for twitter sentiment analysis. *HP Laboratories, Technical Report HPL-2011-89*, 2011.

Zhi-Hua Zhou and Ming Li. Tri-training: Exploiting unlabeled data using three classifiers. *Knowledge and Data Engineering, IEEE Transactions on*, 17(11):1529–1541, 2005.

A Twitrratr wordlist

As the original Twitrratr website has become unavailable, the Twitrratr wordlist could only be retrieved by following links given by a blog author commenting on Twitrratr.[31][32][33] All links were accessed on May the 10th, 2013.

Positives

Woo, quite amazing, thks, looking forward to, damn good, frickin ruled, frickin rules, Way to go, cute, comeback, not suck, prop, kinda impressed, props, come on, congratulation, gtd, proud, thanks, can help, thanks!, pumped, integrate, really like, loves it, yay, amazing, epic flail, flail, good luck, fail, life saver, piece of cake, good thing, hawt, hawtness, highly positive, my hero, yummy, awesome, congrats, would recommend, intellectual vigor, really neat, yay, ftw, I want, best looking, imrpessive, positive, thx, thanks, thank you, endorse, clearly superior, superior, really love, woot, w00t, super, wonderful, leaning towards, rally, incredible, the best, is the best, strong, would love, rally, very quickly, very cool, absolutely love, very exceptional, so proud, funny, recommend, so proud, so great, so cool, cool, wowsers, plus, liked it, make a difference, moves me, inspired, OK, love it, LOL, :), ;), :-), ;-), :D, ;], :], :p, ;p, voting for, great, agreeable, amused, brave, calm, charming, cheerful, comfortable, cooperative, courageous, delightful, determined, eager, elated, enchanting, encouraging, energetic, enthusiastic, excited, exuberant, excellent, I like, fine, fair, faithful, fantastic, fine, friendly, fun, funny, gentle, glorious, good, pretty good, happy, healthy, helpful, high, agile, responsive, hilarious, jolly, joyous, kind, lively, lovely, lucky, nice, nicely, obedient, perfect, pleasant, proud, relieved, silly, smiling, splendid, successful, thankful, thoughtful, victorious, vivacious, witty, wonderful, zealous, zany, rocks, comeback, pleasantly surprised, pleasantly, surprised, love, glad, yum, interesting

Negatives

FTL, irritating, not that good, suck, lying, duplicity, angered, dumbfounding, dumbifying, not as good, not impressed, stomach it, pw, pwns,

[31] http://blog.leenhardt.name/post/2008/10/21/Twitrratr-%3A-how-to-make-a-fuzz-over-nothing
[32] https://docs.google.com/document/d/17rMIILAKsGZAavpipbgQJzdmo9r5SNac6p28S43JrwI/preview
[33] https://docs.google.com/document/d/1rVHVhcC0Eyz4z6TNKXswDh8G7Smw_jDRUNq3R43qZAc/preview

pwnd, pwning, in a bad way, horrifying, wrong, flailing, failing, fallen way behind, fallen behind, lose, fallen, self-deprecating, hunker down, duh, get killed by, got killed by, hated us, only works in safari, must have ie, fuming and frothing, heavy, buggy, unusable, nothing is, is great until, don't support, despise, pos, hindrance, sucks, problems, not working, fuming, annoying, frothing, poorly, headache, completely wrong, sad news, didn't last, lame, pet peeves, pet peeve, can't send, bullshit, fail, so terrible, negative, anooying, an issue, drop dead, trouble, brainwashed, smear, commie, communist, anti-women, WTF, anxiety, STING, nobody spoke, yell, Damn, aren't, anti, i hate, hate, dissapointing, doesn't recommend, the worst, worst, expensive, crap, socialist, won't, wont, :(, :-(, Thanks, smartass, don't like, too bad, frickin, snooty, knee jerk, jerk, reactionist, MUST DIE, no more, hypocrisy, ugly, too slow, not reliable, noise, crappy, horrible, bad quality, angry, annoyed, anxious, arrogant, ashamed, awful, bad, bewildered, blues, bored, clumsy, combative, condemned, confused, crazy, flipped-out, creepy, cruel, dangerous, defeated, defiant, depressed, disgusted, disturbed, dizzy, dull, embarrassed, envious, evil, fierce, foolish, frantic, frightened, grieving, grumpy, helpless, homeless, hungry, hurt, ill, itchy, jealous, jittery, lazy, lonely, mysterious, nasty, rape, naughty, nervous, nutty, obnoxious, outrageous, panicky, fucking up, repulsive, scary, selfish, sore, tense, terrible, testy, thoughtless, tired, troubled, upset, uptight, weary, wicked, worried, is a fool, painful, pain, gross

B Selected word clusters

This section holds some selected Brown word clusters of the clusters by Owoputi et al. (2013), illustrating their usefulness in dealing with spelling variation of words expressing sentiment.

Cluster 1111100110

good gud gd goood gooood goooood qood gooooood goooooood gudd #good goodd guud g00d gooooooood goooooooood gewd -good gooooooooood greaat gooddd gooodd goos qud goooddd go0d gooooooooood g0od goodddd gooooooooooood qudd ghud gooooooooooood goooddd greaaaaat goooodddd goooodd goid guuud gooooooooooooood guddd qoood goooooddd goooodddd //good gooooooooooooood ggod ghudd goooooddd /good

Cluster 11111001010

awesome amazing hilarious brilliant incredible awful awsome priceless unbelievable unreal flawless exhausting hysterical outrageous goed amazin unacceptable unstoppable unforgettable brill breathtaking immense hillarious amazingg astounding orgasmic awesomee appalling atrocious uncanny amazinggg niiiice awesomeee halarious daebak impeccable greatt amzing undeniable awesomesauce amazingggg mindblowing unbelieveable aweful indescribable awsum awesum niiiiice amazayn awesomeeee

Cluster 1111100100100

crazy nuts insane cray tuff krazy crazyy bonkers crazyyy nutz smoove crzy crazyyyy crazzy crazii batshit screwy craaazy crazi #playedout craaaazy brazy liveee crazzzy nutts crazyyyyy cross-eyed faf physco kaput kray reet wackk nutso craazy madt guu liveeee #outrageous crazzzzy craaaaazy dunzo gayyyy crazee crazyyyyyy craz crazzyy greezy wildd buckwild

Cluster 1111001111000

shit ish shyt shxt sh*t shitt sht shiit shidd shiz shittt shizz shii ishh shiiit shitttt s**t shet shiet shittttt shytt glitters ishhh $hit iish shiii sh-t shitttttt issh guap shitz shizzz pu**y isht shxtt shyte sheit bullshitt shieet shxxt sheeit shiiii shit- shitttttttt $h shyd shiitt shiat