Twitter Hashtag Sentiment Analysis

Dinesh Patil
Dept. of Computer Science
University of California, Riverside
Riverside, CA 92521
+1-951-231-7822
dpati003@ucr.edu

Chanapong Thananiwej
Dept. of Computer Science
University of California, Riverside
Riverside, CA 92521
+1-951-379-8592
cthan004@ucr.edu

ABSTRACT

Sentiment analysis as a part of natural language processing deals with understanding the author's opinion. This paper describes a supervised text classification approach to sentiment analysis of Twitter messages known as tweets. Tweets are classified into a category of positive, negative, or neutral according to their overall sentiment with respect to a hashtag. Our system first preprocess tweet to reduce insignificant feature. We trained our classification algorithms including naïve Bayes with available labeled data. This system achieve F1-score of over 0.7 when tested with another labeled tweets resource.

CCS Concepts

• Information systems→ Information Retrieval→ Retrieval tasks and goals→ Sentiment Analysis

• Information systems→ Information Systems Applications→ Data Mining.

Keywords

Sentiment Analysis; Twitter Hashtag; Data Mining; Data Crawling; Classification models; Naïve Bayes; Tweet

1. INTRODUCTION

Over the past decade social networking sites have become an important platform of social interactions and knowledge sharing among individuals throughout the world. In the world dominated by social media, individuals as authors of social media posts and comments have obtained a lot of power to describe their opinions and experiences. Besides a social platform, it also forms a juncture of social marketing for businesses to interact with and acquire customers. Businesses can be greatly affected by the opinions of individuals, as it is broadcasted over theses social platforms. It thus becomes vital for businesses to understand the sentiment of the majority of customers or probable customers, to realize the type of opinions regarding their products as well as gain insight about the customer needs and complaints.

Twitter is a very powerful means of conveying opinions and garners a lot of public attention. Twitter data is very well suited for opinion mining as it has a limited size of 140 characters and thus tends to have specific sentiments. Besides that, the hashtag feature allows searching for tweets related to a specific topic. The above properties make twitter data good candidate for sentiment analysis.

The aim of this project is to collect twitter data specific to a topic using hashtag search based web crawler and mine and analyze the data to realize the possibility of predicting sentiments of the tweet.

The motivation behind using hashtag for data crawling is:

a) Hashtag is very specific to a topic.

   For example, if we were to search for tweets with respect to an entity keyword or username such as McDonald's, we are bound to get a lot of tweet with various topics regarding the user/entity. However, if we search based on a specific hashtag such as #McDStories, we get tweets specific to experiences people have shared related to their visits to McDonald's.

b) Often, hashtags are used by companies and advertising firms to promote a product or service. If the companies want to analyze the effectiveness the success of the campaign hashtag based sentiment analysis will help them understand the trend of the public opinion about the product/service.

The project aims to develop an architecture to process this twitter data to mine sentiment and opinion knowledge. The architecture can be divided into 4 major parts:

(1) Data acquisition using API

(2) Acquiring sample training set and generation of a sentiment dictionary

(3) Preprocessing the data

(4) Applying classification algorithms to classify test data into 3 sentiments (positive, negative and neutral).

The project is however not only limited to this architecture as several post-processing steps are required to obtain pattern knowledge and probable causality of the results. Classification results will be validated using parameters such as precision and recall. Data visualization also forms an integral part of the project to show proof of results and acquire visual representations of the knowledge.

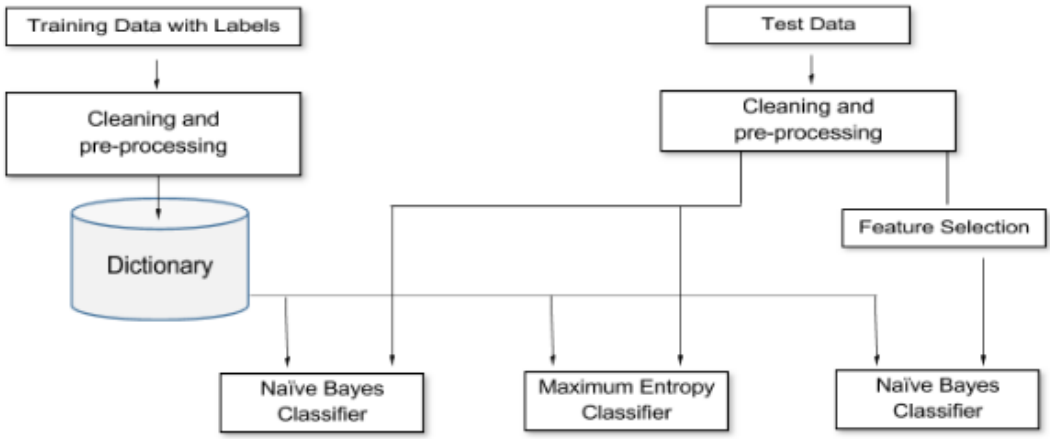

Figure 1. System Architecture

The techniques used for the classification of tweets into the three sentiment classes are:

a. Naïve Bayes Classifier:

   Naïve Bayes is basically a simple probabilistic supervised classifier based on applying Bayes' theorem with strong (naive) independence assumptions between the features.

b. Naïve Bayes Classifier with Sentiment Lexicon:

   Sentiment lexicon is a dictionary that gives higher weightages to some keywords.

Once we have classified the data (tweets) we can extract the following information from it:

a. Ratio of positive, neutral and negative tweets for a specific hashtag.

b. Top positive and negative tweets

c. Most common keywords in each class of tweet.

d. Trend of the tweets sentiment over a specific timeline.

This information helps us analyze the sentiment regarding the topic/hashtag and not only decipher the common reasons for the sentiment but also plot its trend along a timeline and/or location.

2. RELATED WORK

2.1 SWASH: A Naive Bayes Classifier for Tweet Sentiment Identification

In "SWASH: A Naive Bayes Classifier for Tweet Sentiment Identification," the authors explain the usage of naïve Bayes classifier in sentiment analysis of tweets. Their system uses both training dataset of labeled tweets and external sentiment lexicon. During the preprocessing steps, the program tokenizes and normalizes text data as well as negates the sentiment of text when negate keywords are discovered. It also uses Laplace smoothing when encountering a new word and try many other techniques, but unfortunately they do not improve the F1-score, and some techniques even make the system performs worse. The system was submitted to SemEval-2015 competition in sentiment analysis and rank 21st out of 40.

The paper shows that using external sentiment lexicon gives highest improvement of F1-score, increasing it by 5.81. In addition, assigning higher weights of 5 times to lexicon than training dataset improves F1-score as well. This is because available sentiment lexicon are trained by huge amounts of data and therefore has more accurate sentiment probability. Other techniques employed that increases score are lowercasing and overriding. Overriding refers to a rule-based system that overrides the naïve Bayes output if certain emoticons and words are found. Overall, the authors show that preprocessing text and assigning weights significantly improve the performance of naïve Bayes.

On the other hand, the authors outline that other classifiers including LinearSVC and NearestCentroid were tested. However none can outperform naïve Bayes. But, the program uses the same preprocessing techniques and feature selection for other classifiers as naïve Bayes, which may affect performance. The authors also mention that employing too many techniques lead to overfitting the data. For example, using spell corrector results in worse F1-score.

A potential extension to this system is giving higher sentiment weight to capitalize words. In the preprocessing step, the program lowercases all words such as "Hello," "HeLlo," and "HELLO." While the first two cases are from capitalizing the first letter in a sentence or typo, the last case of capitalizing all letters often means that stronger sentiment is putting into this word. As such, assigning higher weights may improve performance.

2.2 Twitter Sentiment Classification using Distant Supervision

In this paper, the authors describe the usage of three classifiers of naïve Bayes, maximum entropy, and support vector machines to classify sentiment. To train these classifiers, the program uses distant supervision, in which the program automatically label tweets with emoticons instead of manually labeling. Training data has positive sentiment if it contains positive emoticon and negative for negative emoticon. Standard preprocessing techniques of text data such as n-gram models are used. This particular system shows that maximum entropy performs better than the other two classifiers.

Two interesting approaches of sentiment analysis that this paper explains are query term normalization and usage of both unigram and bigram at the same time. Query term can be positive or negative, but it should not affect the overall sentiment. For example,

if the query term is "happy," then the result will mostly be positively bias by the sentiment of the query term. The second technique of using both unigram and bigram improves the performance of naïve Bayes and maximum entropy.

An important limitation of this system results from the use of distant supervision. Since emoticons are used to label training dataset, all training tweets contain emoticons. Consequently emoticons have a larger weight than the text and dominate the sentiment result. But, not all tweets with positive emoticons have positive sentiment overall. As such emoticons are stripped from the classifier algorithms.

One possible extension is limiting the classifier to certain topics. A word can have different meaning and thus different sentiment in different topics. Therefore focusing on certain topics may lead to more stable sentiment probability. Another extension is removing the limitation and use emoticons in the algorithm.

2.3 Using Maximum Entropy for Text Classification

The paper aims at introducing maximum entropy models for text classification. Through the course of the paper, the concept of maximum entropy is explained along with the reasons explaining why maximum entropy is a good candidate for text classification compared to various other classification models, especially Naïve Bayes and scaled Naïve Bayes.

The authors define the principle idea of maximum entropy classifications as '… without external knowledge, one should prefer distributions that are uniform….'. In this paper, maximum entropy is used for text classification by estimating the conditional distribution of the class variable given the document. The authors demonstrate the use of Improved Iterative Scaling (IIS) and Gaussian Prior for improvement of a regular maximum entropy classifier.

In general, maximum entropy can be used to estimate any probability distribution, but the authors limit their model to classification and hence the discussion aims at learning the conditional probability distribution of the class label given a document.

The paper also provides a solution to the overfitting problem of maximum entropy, using a Gaussian prior. The prior is integrated using a posteriori estimation for the exponential model, instead of a maximum likelihood estimation. This prior favours feature weightings that are closer to zero, that is, are less extreme thus controlling overfitting.

In the results section, the authors compare the performance of maximum entropy models with and without priors against regular naïve Bayes and scaled naïve Bayes classifiers. It is demonstrated that in most of the datasets, maximum entropy provided better results and even better results with the use of the prior. However, it is noticed that improper feature selection causes severe degradation for the maximum entropy classifiers.

Areas of improvement mentioned include adjusting the prior based on the amount of training data, augmenting maximum entropy with expanded feature classes and thorough comparison between state of the art classifiers for various domains to obtain better insight into the workings of the classifiers.

2.4 Sentiment Identification Using Maximum Entropy Analysis of Movie Reviews

The paper aims at using classification algorithms to analyse sentiments specific to user's preference. The paper also explains the importance of finding user specific classification as opposed to generic opinion mining, as it provides much better accuracy in suggestion applications. It also explains how a machine learning based approach proves better when dealing with a lot of data (reviews) and is independent of biases as opposed to human classifications.

The paper also provides a brief comparison of Naïve Bayes, K-nearest neighbour and Maximum entropy models, each with the assumptions, pros and cons. Followed by this the explanation of maximum entropy model used by the authors to classify movie reviews using sentiment analysis. A detailed overview of the maximum entropy classifier ,steps of data acquisition, pre-processing and the types of classifier model combinations used are explained further.

The principle property of maximum entropy technique is defined by the authors as '… provides a probability distribution that is as close to the uniform as possible given that the distribution satisfies certain constraints… 'The words, which serve as features for a text are chosen using the chi-square method of Chi-square selects words that have the most skewed distribution across categories.

The future work proposed by the authors includes recognition of semantic and linguistic features instead of just the statistical significance of words and implementation of Improved Iterative Scaling (IIS), to improve performance.

3. PROPOSED METHOD

We get Twitter data by implementing a crawler to crawl tweets from the Twitter API. We used external resources of labeled tweets for the purposes of training and testing. Our system used many preprocessing techniques and two classifiers including naïve Bayes baseline and with sentiment lexicon. This section describes our approach in details.

3.1 Data Acquisition

Twitter Application Programming Interface (API)[1] provides developer with many ways to access Twitter data. To use the API, we first need to create a Twitter developer account. In addition, twitter uses OAuth[2] to provide authorized access to its API. OAuth is the authorization framework that enables an application to obtain limited access from an HTTP service. As such, we use 2 external python library, httplib2 and python-oauth2[3], to access Twitter API. We use REST API of endpoint GET search/tweets to search tweets by a hashtag. This connection point has a rate limit of 180 requests per 15 minutes. Twitter data is provided in JSON format. Our crawler then parses the JSON data and gets Tweet messages and the date. Table 1 shows a few examples of the crawler output. Due to the limitation of Twitter API which only gives data of one week before the crawl date, we only manage to get data in November.

[1] More information about the Twitter API can be found from the link https://dev.twitter.com/docs.

[2] More information about OAuth can be found from the link https://oauth.net

[3] Httplib2 and python-oauth2 can be download from the links https://pypi.python.org/pypi/httplib2 and https://github.com/joestump/python-oauth2

Table 1. Example of Twitter API Data

Hashtag	Tweet
Hurricane Matthew	It's been over a month since #HurricaneMatthew and I still can't see my desk because of all the files
	#Flood Proof of Loss Extension Granted for #HurricaneMatthew Victimes https://t.co/tqGZIVPGWl
	Thinking about how to donate this #GivingTuesday? Consider the ongoing #HurricaneMatthew recorvery: https://t.co/9fLG2bOy2e vie @TiKayHaiti
Trump	The @realDonaldTrump is draining the swamp into the white house @Lawrence @maddow #Trump #msnbc #LoserDonald
	@CommonSenseHour @HuffPostPol #Trump (sniff) may not be too happy (sniff-sniff). Feel a 3am rant coming on as a result.
	#ElectoralCollege either vote to keep #trump out because thats their design or they r #irrelevant and next election #outofwork

3.2 Training Data

There are many resources of labeled tweets available online. For this project, we use Twitter data obtained from https://github.com/ravikiranj/twitter-sentiment-analyzer/tree/master/data/test. We used this resource because the author does not preprocess tweet message, and the format is very clean. Table 2 show some examples.

Table 2. Example of Training Data

Sentiment	Tweet
positive	Thanks for pointing out the crucial problems @thakkar. Both of them have been taken care of (cc: @Netra)
positive	just got home from a meeting with the girls... Maaaaaan I'm exhausted!! Goodnight world
negative	Can't believe I have to wait another 6 months for my phone contract to end! I'm bored now!!! The 12 month contract would have run out!!!
negative	ugh. a huge headache, coughing constantly, legs feeling week, and feeling like throwing up. This sucks beyond compare
neutral	Hey @apple, androids releasing brand new state of the art phones, whens your new phone come out? What's that? (cont) http://t.co/2sko9l3d

3.3 Preprocess

In natural language processing, preprocess serves the following 2 purposes:

1. Removes irrelevant features that do not contribute to the analysis of the language
2. Strengthens the weights of relevant features by grouping similar features together

For sentiment analysis, irrelevant features are those features that do not change the overall sentiment of the tweet messages. Thus, we try to remove them as much as possible. On the other hand, relevant features are those that do change the sentiment. Since our proposed classifiers are probabilistic classifiers, we group words with the same sentiment as much as possible in order to increase the probability of each words, which in turn increases the overall accuracy.

As prior related works of sentiment analysis had shown, preprocessing text data to reduce insignificant features improve accuracy immensely. Therefore, several preprocessing techniques were considered, but not all were used. We discuss the decision of which techniques to use below. Table 3 shows the change in average F1-score for each preprocessing techniques.

Table 3. Effects of Preprocessing Techniques

Preprocessing Technique	Average F1-Score Difference
Normalization	-0.016
Remove Stopwords	0.004
Remove URL	0.001
Remove Username	0.000
Negation	0.003

3.3.1 Normalization

We lowercase all tweets for the purpose of grouping words together (e.g. "hEllO" to "hello"). However, this preprocessing step reduces the F1-score. Further analysis shows that uppercase and lowercase words seem to express different sentiment. For example, "HATE" has a stronger negative sentiment than its lowercase counterpart of "hate". Thus, we decide not to normalize tweet and leave the capitalization as is.

3.3.2 Feature Extraction

Similar to other natural language processing applications, we remove stopwords as they do not affect sentiment (e.g. "and", "I", etc.).

In addition to stopwords, tweet data has unique features that do not contribute to the overall sentiment of the messages, thus, they were removed. These include:

1. **URL** Author has an option to include links in a tweet. Twitter has a unique encoding of the URL that begin with "https://t". We remove these links.
2. **Username** Username is Twitter account name that the users can set and change to any text. Twitter used "@"

symbol to mark the start of username. Since it can be changed to anything, we considered it noisy data and removed it (e.g. @IAmAUser).

3.3.3 Negation

In natural language processing, the words "no" and "not" change the sentiment of the next word to the opposite. Therefore, we negated it by adding "NOT" to the next word, and now this word is considered as a new word (e.g. "not good" becomes "NOTgood", "not bad" becomes "NOTbad").

3.3.4 Tokenization

We tokenize the text data into a bag-of-words, unigram model in order to calculate probability and build the sentiment dictionary. Unigram also complements naïve Bayes well because naïve Bayes assumes that each words is independent of one another.

3.3.5 Other Preprocessing Techniques

Other preprocessing techniques were considered such as stemming and lemmatization. Due to the casual nature of tweets however, we decided against grouping words with similar meaning into a single group because they sometimes represent different sentiment. This is similar to the negative F1-score difference we got when we do normalization. For example, "hated" is a verb that can have a weaker negative sentiment than its present tense version of "hate" as the past tense may implies that this person does not hate the other person anymore in the present.

3.4 Classifiers

We choose naïve Bayes because it is a widely used text mining algorithm due to its high accuracy. While the algorithm is not complex and its assumption of independent features are wrong for text data, naïve Bayes works surprisingly well in practice as prior related works had shown. Two versions of naïve Bayes is implemented so that we can compare and analyze the accuracy between them. As a baseline, we used naïve Bayes from an open source python library NLTK. We then implement our own version of naïve Bayes with sentiment lexicon and hope that we will achieve higher accuracy than the baseline.

3.4.1 Naïve Bayes Baseline

We used naïve Bayes from the NLTK library as our baseline. NLTK is an open source python library that has many data mining algorithms such as feature selection and classifiers.

3.4.2 Naïve Bayes with Sentiment Lexicon

We implement our own naïve Bayes classification model that incorporated an external sentiment lexicon to increase our accuracy. Sentiment lexicon is a dictionary that consists of words that humans rated as either strongly positive or negative. The lexicon we used gives strong sentiment word a maximum of 4 times the normal weight.

With sentiment lexicon, naïve Bayes formula becomes:

$$P(C_i \mid Tweet) = P(C_i) \prod w_k \times P(word_k \mid C_i)$$

where C_i is either the class of positive, negative, or neutral and w_k is the weight provides by sentiment lexicon

4. EXPERIMENTAL EVALUATION

We evaluate our classifier models by using a random subset of our training data for testing. The testing data samples are randomly sample to have the same distribution of classes as the training data. We use the same amount of preprocessed training and testing data on both algorithms to fairly evaluate.

4.1 Naïve Bayes Baseline

The NLTK library has its own testing function that takes a labeled testing data and output its accuracy. The baseline manages to achieve an accuracy of 0.861.

4.2 Naïve Bayes with Sentiment Lexicon

Without sentiment lexicon, the average F1-score is 0.723. With the lexicon, the average increases to 0.727. Interestingly, the lexicon only increases the F1-score of positive and negative case. This may be because the lexicon only assign higher weights to positive and negative words.

Table 4 shows the baseline predicted outcomes versus expected outcomes.

Table 4. Sentiment lexicon results

	Predicted Positive	**Predicted Negative**	**Predicted Neutral**
Expected Positive	3227	1104	5
Expected Negative	554	3781	1
Expected Neutral	633	500	1138

Table 5 shows precision and recall as well as F1-score of the baseline algorithm for each sentiment classes. F1-score is the harmonic mean of precision and recall.

Table 5. Sentiment lexicon evaluation

Sentiment	**Precision**	**Recall**	**F1-Score**
Positive	0.744	0.731	0.738
Negative	0.872	0.702	0.778
Neutral	0.501	0.995	0.666

Table 6 is the accuracy and the average F1-score of each classes.

Table 6. Sentiment lexicon accuracy and average F1-score

Accuracy	**Average F1-score**
0.740	0.727

5. DISCUSSION

Unfortunately, our system is unable to achieve a higher accuracy than the baseline. This may be because the NLTK library has its own training algorithm that is carefully designed to get the most accuracy possible.

Table 5 shows one interesting result of our system. The recall of neutral class is very high however, the F1-score is offset by the low precision. A precision of 0.5 in the neutral case means that many expected neutral tweets were misclassified as positive or negative.

In order to see our results better, we created a few data visualization.

1. Word Cloud

Word cloud shows words with high frequency of each sentiment classes. It is an amazing representation that helps us analyze most frequent words associate with a hashtag. Figure 2, 3, and 4 show positive, negative, and neutral word cloud of the hashtag #HurricaneMatthew.

Figure 2. Positive Word Cloud of #HurricaneMatthew

Figure 3. Negative Word Cloud of #HurricaneMatthew

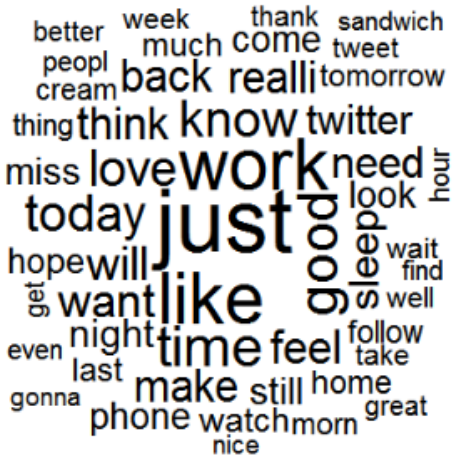

Figure 4. Neutral Word Cloud of #HurricaneMatthew

2. Timeline

Timeline provides us with the sentiment distribution. We analyze the distribution in months. We can now see the trend of what sentiment does the hashtag starts with and how it changes overtime. Due to the limitation of Twitter API which only gives data of one week before the crawl date, we cannot create a timeline for the whole year. Figure 5 is an example of the timeline for positive sentiment we can create if we have the data.

3. Percentage of Positive vs Negative vs Neutral

Knowing percentage of each sentiment classes provide us with the overall sentiment of the hashtag. We used this percentage to further prove that our experimental evaluation is accurate by checking the testing data for percentage of each sentiment classes. Figure 6 is an example pie chart of the percentage.

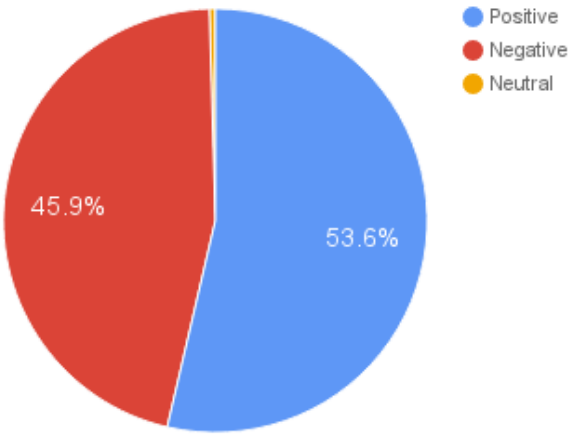

Figure 6. Sentiment Percentage of #Trump

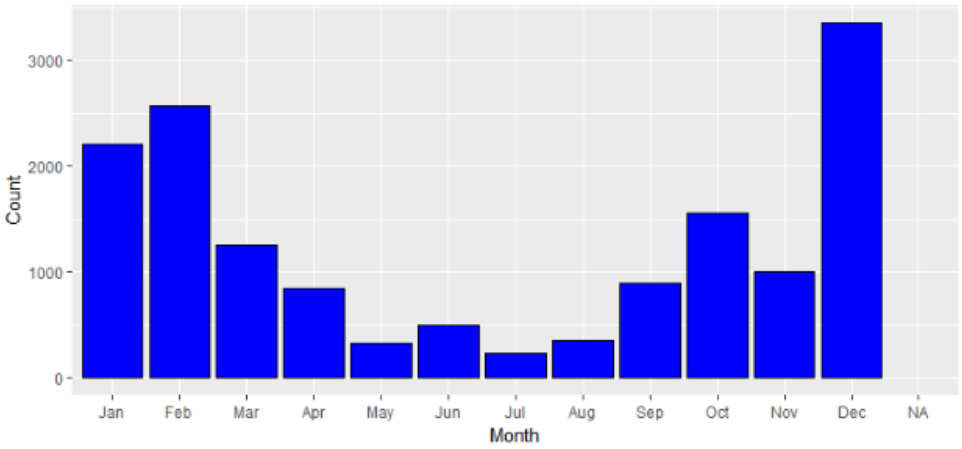

Figure 5. Positive Timeline of #HurricaneMatthew

4. Top Positive and Negative Tweets

Top positive and negative tweets, unlike word cloud, provide us with real tweet messages. We can learn, in general, the sentiment of each words in the tweets. For example, the most negative tweets will consist of mostly words with high probability of negative sentiment. This apply to both positive and neutral case as well. Table 7 shows top 3 of most negative and most positive tweets about #Trump.

Table 7. Top 3 of Most Positive and Negative Tweets of

Sentiment	Tweet
Positive	https://t.co/IhDKRTVNkf #PodcastMotorLat podcasting #fun #Trump #Opinion #girlsNightOut #blessed #blesser #blesse
	Wow, wow, wow. Adolf approves #UnitedStatesOfTrump #Trump proposes stripping citizenship from political protesters https://t.co/zEtxJoa9eV
	https://t.co/OG1SS1HWG6 Daily! #Liberty #Freedom https://t.co/lXtRL6QtcZ Thanks to @JeffreyCOwnby @ZiffyKat @CapeFlo #2a #trump
Negative	560 #lies: LIARS, DAMN LIARS, POLITICAL LIARS & #TRUMP - RIGGE… https://t.co/ewwIABr6rv
	Arrogant, haughty, conceited, egotistical, bumptious, overweening, big-headed, pompous, imperious, janky, triflin s… https://t.co/niX1hfvl8c
	@MariaTCardona @PVL955 @CNN haters gonna hate #Trump deplorable

6. CONCLUSION

This paper explains a sentiment analysis system that emphasizes preprocessing techniques and naïve Bayes classifier. We shows that naïve Bayes is an impressive classifier for sentiment analysis. The naïve Bayes baseline from the NLTK library achieves 86.1% accuracy while our own implementation using sentiment lexicon achieves 74% and an average F1-score of 0.727. We feel that we can increase our accuracy with more training data and lexicon.

7. FUTURE WORK

- **Sarcasm** In natural language processing, sarcasm refers to text data with multiple negative sentiment words, but should be classified as positive sentiment or vice versa. Detecting sarcasm will immensely help lower the false positive and false negative rate.
- **Maximum Entropy** Since maximum entropy does not assume independency between features, we may be able to increase the accuracy of our system. Prior related works also show that maximum entropy is a superior sentiment analysis algorithm.
- **Impact of Capitalized Letter** Currently, our system leaves capitalized words as is. Because of the casualness of tweet messages however, capitalized words often have more impact thus should be given higher weights. This is similar to the combination of normalization preprocessing technique and sentiment lexicon.
- **Map Visualization** In order to improve our data visualization, we plan to create a map of tweet messages location. We can then plot the location on Google maps and gives different colors to each sentiment classes. This plan was scrapped however because not many tweets contain location and due to the limited amount of data we can get from the Twitter API

8. ACKNOWLEDGEMENT

Our thanks to Professor Vagelis Papalexakis for teaching us about data mining as well as guiding us about the project. We also extend our thanks to Ravikiran Janardhana for providing us with an online resource of labeled tweets.

9. REFERENCES

1. Go, Alec, Richa Bhayani, and Lei Huang. "Twitter sentiment classification using distant supervision." *CS224N Project Report, Stanford* 1 (2009): 12.

2. Talbot, Ruth, Chloe Acheampong, and Richard Wicentowski. "SWASH: A Naive Bayes Classifier for Tweet Sentiment Identification." *SemEval-2015* (2015): 626.

3. Mehra, N., Khandelwal, S. and Patel, P., "Sentiment Identification Using Maximum Entropy Analysis of Movie Reviews", *Stanford University, USA in 2002.*

4. K. Nigam, J. Lafferty, and A. McCallum. Using maximum entropy for text classification. *IJCAI-99*

5. Workshop on Machine Learning for Information Filtering, pages 61–67, 1999.