In Proceedings of the 7th ACM SIGSPATIAL Workshop on Location Based Social Networks (LBSN'14), Dallas, TX, November 2014, pp. 49-56.

A Look into Twitter Hashtag Discovery and Generation[1]

Eric Krokos Hanan Samet Jagan Sankaranarayanan
Department of Computer Science
University of Maryland, College Park
College Park, MD 20740
{ekrokos,hjs,jagan}@umiacs.umd.edu

ABSTRACT

Twitter is a micro-blogging site that allows users and companies to post brief pieces of information called "Tweets". Some of the tweets contain keywords such as "Hashtags" denoted with a "#", essentially one word summaries of either the topic or emotion of the tweet. The goal of this paper is to examine an approach to perform hashtag discovery on Twitter posts that do not contain user labeled hashtags. The process described in this paper is geared to be as automatic as possible, taking advantage of web information, sentiment analysis, geographic location, basic filtering and classification processes, to generate hashtags for tweets. Hashtags provide users and search queries a fast and simple basis to filter and find information that they are interested in.

Categories and Subject Descriptors

H.3 [Information Storage and Retrieval]: Information Storage and Retrieval

General Terms

Algorithms, Design

Keywords

Twitter, News, Hashtags, Topic Classification, Sentiment Analysis, Web, Tweet, Natural Language Processing

[1] This work was supported in part by the NSF under Grants IIS-10-18475, IIS-12-19023, and IIS-13-20791 and by Google Research and NVIDIA.

ACM SIGSPATIAL LBSN'14, November 4, 2014, Dallas, TX, USA © 2014 ACM. ISBN 978-1-4503-3140-1/14/11 ... $15.00
http://dx.doi.org/10.1145/2755492.2755500

1. INTRODUCTION

1.1 Motivation

The use of Twitter has exploded over recent years along with other micro-blogging sites. Twitter allows users to post 140-character "Tweets" that they share with other users. The tweets can range from topics about a user's opinion, their daily life, a new product being released, or even the News. In the last few years, social media has greatly changed the way news is gathered and reported. In particular, in the Egyptian 2011 revolution, Twitter and other social media services were the primary methods that reporters and news agencies reported on the unfolding events [11]. According to Twitter, they receive roughly 500 million tweets per day, or about 5,700 per second, as of the writing of this paper in 2013 [8]. With this amount of data being generated each day, it's not a surprise that scientists and news reporters alike are interested in looking into this data. However, filtering through the data to find what you want is a huge problem. Hashtags provide a shortcut in filtering, in that users can query on certain hashtags and get tweets labeled with that hashtag. Unfortunately, most tweets do not contain hashtags and this makes filtering and searching for information more difficult.

1.2 Limitations

The huge corpus of data generated by Twitter users presents many challenges. The nature of computer chat makes using Twitter data very difficult. Online society and language is very different from normal everyday language in that the rules of grammar and spelling don't necessarily apply. The fact that Twitter enforces a 140-character limit on tweets forces users to be creative in how they write their posts. Instead of writing out a whole name or word, users will often abbreviate or just leave out words all together. The number of different ways users can write words or misspell them creates a feature explosion problem. For example, the presented algorithm has counted 12 different ways of spelling Obama, with variations of capitalization, exaggeration, spellings, and the lacking of spaces between words. Another problem is the unique language that has developed on the internet, with words and groupings of words being converted to shorthand such as "Laugh out loud" represented as "lol" or "glhf" as "Good luck, have fun". There are many different examples of this and many people have their own unique variations of this shorthand.

If a user wants to post more than the 140-character limit, Twitter does allow the use of web links within a tweet, allowing a viewer to click the link and get more information elsewhere. These links can go to anywhere on the internet, posing interesting issues. First, many links only exist for a certain period of time, after which they expire and the link is no longer valid. Another problem is that certain links may take you to harmful websites.

1.3 The Approach

The goal of this paper is to present a method to label and generate hashtags for tweets that do not have hashtags. By labeling tweets with hashtags it allows the filtering and searching of data to be much easier. The first task was to gather a large collection of tweets. A simple Java program using the Twitter4J library was used to log into Twitter and collect tweets from the Twitter stream. In all, it gathered roughly 150,000 tweets over the course of a few months with the requirement the tweets be in English.

After gathering a large amount of Twitter data, the first thing to do was to analyze and format the data. The first thing done was check to see if there were any links and visit them. The Java web package was used to visit the links and retrieve the html data. The idea is that we want to expand upon the information given in the tweet with the web data. Given the expanded set of web data, we run a spell checker on the text. The purpose is to consolidate the different spellings of the same word. Now we remove unimportant words such as "the", "to", "and", to reduce the word feature space. After, we generate word stem-pairs, in particular 2 and 3 pair. This is so that if the words Barack and Obama both occur in a sentence, a word called BarackObama will be created. Next, to make sure the algorithm is not cheating, any hashtags included in the raw tweets are removed. Once the hashtags have been removed, we perform a basic sentiment analysis that looks for keywords or expressions (such as emoticons) in a tweet and assigns 1 of 18 emotions to that tweet. If the tweet contains a hashtag, then the same emotion previously assigned to the tweet is assigned to the hashtag. Each tweet is collected with a latitude and longitude pair and is mapped to one of the top thousand most populated cities and countries. This is allows us to filter on hashtags and tweets based on their locations.

Now that the Twitter data has been filtered and configured, we need to calculate the importance of each word. The TF-IDF of each word is calculated for each Tweet. If we are generating hashtags that didn't exist in the initial corpus, the most important word (the highest score) is chosen as a hashtag for the tweet. Then a feature vector, where each dimension is a word, is created for each Tweet. Additional features are included for the nearest major city to the location of the tweet, which allows the system to incorporate the geographic location of the tweet as context. A classifier is trained on the feature vector and is used to match each tweet with at least one hashtag.

1.4 Evaluation

Evaluation of the success or failure of the algorithm is difficult because of the lack of ground truth. What we do know is that certain tweets have hashtags associated with them. If the algorithm is able to generate the same hashtag for a tweet, even with the hashtag label removed from within the tweet, then that is considered a success. After generating hashtags for the tweets, a human interpretation of the correctness of the hashtag assignments is still required.

1.5 Paper Structure

The rest of the paper is organized as follows: Section 1 reviews previous work on topic classification and hashtag discovery. Section 2 discusses goals and approaches, while Sections 3 and 4 presents an evaluation of the work done and how the results are generated. Section 5 provides concluding remarks.

2. BACKGROUND AND RELATED WORK

2.1 Topic Classification

Topic Classification is a mature field in Natural Language Processing (NLP). It involves a corpus of documents with each word in the document being represented as a multi-dimensional vector. The more words in the documents, the higher the dimensionality of the vectors. A common example of Topic Classification is the Spam / Not Spam problem. The idea is, given a document, to classify it as Spam or Not Spam. Starting with a large collection of pre-labeled documents, feature vectors are constructed for each in a preprocessing phase. Typically, a variant of Term Frequency – Inverse Document Frequency (TF-IDF) is used. Essentially, if a word is common in a document or is common across all documents, it is given a very low value, signifying low importance. If a word occurs rarely in a document or rarely across all documents, it's typically given a high score. Another example of Topic Classification is the multi-class classification. This type of classification is used when the document type span more than 2 topics. For example, the documents you want to classify may have the topics, Action, Adventure, Sci-Fi, Mystery, etc. There are many classifiers to choose from to handle multi-class classification; they range from simple clustering algorithms such as K-Means or K-Nearest-Neighbor to more complex ones such as Naïve Bayes or Support Vector Machines (SVMs). Once the classifier and type of classification are selected, a training and testing set of data are generated, typically dividing the data corpus into 70% for training, 30% for testing. You train your classifier on the training set, and then evaluate its performance on the testing set.

2.2 Twitter work and Sentiment Analysis

While it appears there is little work done on Twitter hashtag discovery, there has been plenty of work using Twitter data [4, 11, 13]. For example, there's been work on doing real-time event detection using Twitter data [6, 12, 19]. The idea is to be able to figure out if an event is occurring and any other additional information based on different attributes of twitter posts. Other work has been done on the sentiment analysis of tweets [1]. The idea behind this work is they have a large dataset of words, hashtags, patterns of words, and emoticons or smiles associated with a set of emotions. Using this dataset, the authors go through each tweet and each word and see if there is a matching between the word and a sentiment in the dataset.

3. GOALS AND APPROACHES

3.1 Tweet Gathering

The Twitter data was gathered using a program written in Java using the Twitter4j library. The library requires a Consumer key and secret key, along with an access token and token secret. A twitter stream is set up that collects tweets from a pre-selected set of users. When a tweet is received, it contains the user name, user id, raw tweet information, any hashtags associated with it, and other information. Once the information we're interested in is gathered, it's stored for use at a later time. The system was setup to run for 1 hour intervals in case of connectivity loss. Overall it collected 150,000 tweets over the course of a few months.

3.2 Web Gathering

Twitter allows the addition of links into a tweet. For us this is a huge benefit because it allows more data to be associated with a

potential hashtag beyond the normal 140 character limit. First, the algorithm has to determine if a tweet contains a link. Fortunately, the links all start with "http" or "https" which makes finding the location of the link easier. Once a tweet is known to have a link in it, the Java URL library is employed to follow that link. What we want to do is expand the information given in the tweet by appending it with text from the link, so what we're interested in is everything within the paragraph brackets of the html ( ). The algorithm gets all the information between all the paragraph brackets and then continues to process it. This strategy has been used in the TwitterStand [3, 6, 19] system which looks for newsworthy tweets from reliable tweeters and associates them with a map at the location that serves as the geographic focus of the tweet. The results are displayed on a map and are part of our work on developing spatial browsers for spatial data that is expressed geometrically with lat-long coordinate values (e.g., QUILT [16, 20] and the SAND Browser [2, 15]) and textually (e.g., STEWARD [9], NewsStand [14, 17, 18, 21]).

One lesson learned during this research is that adequate anti-virus and internet security software is important when downloading content linked from tweets. While gathering data from the web, a couple of tweets had links that contained websites with viruses and other forms of malicious software.

For each block of text that was gathered from the web, a count was kept of the number of times that exact block of text was encountered. The reason for this was if a link or page was broken, the connection would often return the same or similar error. For example, these are some of the errors or blocks of text returned and the number of times they were encountered.

- (232) FORGOT YOUR ?
- (116) ADD THIS TWEET TO YOUR WEBSITE BY COPYING THE CODE BELOW.
- (116) HMM, THERE WAS A PROBLEM REACHING THE SERVER.
- (132) FOR YOUR SMART PHONE AND START EXPLORING THE WORLD AROUND YOU!

The web data for each tweet was appended to its raw tweet only after all the web data for all tweets was gathered. Once the histogram of the number of times each text block was encountered is filled, then the appropriate web data associated with each tweet is appended. The reasoning behind having this system is that we don't want to add information to a tweet that is common across many tweets or common across only the tweets with links. Also, since it's hard to know if a link is broken, this provides a "learned" way of finding out. For the purposes of this algorithm, if a block of text occurred more than 5 times, it's considered spam or unimportant text.

3.3 Spell Checking

The tweets gathered were filtered so that the algorithm only kept the ones that were in English. Given that, performing spell check on the words gathered from Twitter is still an extremely difficult task because of the unique way the internet uses language. Twitter forces uses to conform to a 140-character limit on their messages. Users get around this by using a variety of tricks such as using shorthand or purposely leaving out letters in words while leaving the main idea intact. Another difficulty is the use of symbols in the messages. To emphasize certain words or to portray emotional information, a user may put emoticons or other symbols in their messages.

The first step is to remove the symbols and emoticons from the text. This is a relatively simply process in Java as all we need to do is to make sure all the characters within the tweet are alphabetical or numerical, and remove all those that are not. From there, a spell checker is employed. A large file of words is loaded into a library called Jazzy. Given a word and a number of steps, the library will modify the given word within n steps to match a word within its database. The higher the value of n, the more "adventurous" the spell checker.

The main problem faced with this is the fact that many words used on Twitter are not actually words. For example, "Lol" and "glhf" are not words. Without adding those words into the spell checker, the library would attempt to correct those words. For example, before adding "Lol" into the database, "Lol" would be corrected to "Look" if n was set to 2. What was done was different words used on the Internet from various websites were gathered and added into the dictionary. All of the words added were screened by a human. Initially this process was going to be done automatically, by seeing how many times a unique word occurs. For example, if the word "Lol" occurred more than 5 times, then we would say it's a word. The problem with this is that when you encounter "Oboma" rather than "Obama". The word "Oboma" was found 23 times in a corpus of 150,000 tweets. "Oboma" is not a word, and we want to make sure "Oboma" is corrected to "Obama". It would be a challenge to figure out at what threshold to accept a word.

3.4 Stemming

Word stemming was also considered but later rejected. The primary purpose for spell checking is to reduce the word dimensionality by fixing misspelled words so that they can be compared correctly. Stemming would not solve this, for example, if you set the stem to be "Oba", "Oboma" and Obama would not be in the same stem, unless the stem was of length 2 to generate the stem "Ob". Of course, by making the stem this length, you allow many other words to fit into the stem "Ob". While this would reduce the dimensionality of the words, it removes crucial details from an already restricted dataset (the 140-character limit). This sentence stemming process were employed with all tests and not evaluated on its own.

3.5 Sentence Stemming

Even though the notion of word stemming was rejected, sentence stemming is very important. Sentence stemming is the process of taking 2 or more words in a tweet, and combining them into a single word. The idea is that if two words occur in a tweet, and they are mutually important, then it's important to make that a feature. For example, if Obama and Washington occur, then a new word called ObamaWashington is added to the tweet. We want to emphasize that these two words occurred together, rather than Obama and Hillary. The idea is this that can filter or encourage certain hashtags to be associated with the tweets. For the purposes of this paper, length 2 and 3 n-grams of the tweets were used. The n-gram computation was performed on the raw tweet and any web data gathered.

This process exploded the number of features, rather than having a feature for every word, now there is a feature for every pair and triple of words. The hope was that while making everything more computationally expensive, it would give key insight into important features in a tweet.

3.6 Geo-Location

An important component of a tweet's context is geographic location. When the tweets were gathered, the latitude and

longitude of the tweet were recorded. The idea is that the physical location of where a tweet originates may prove helpful in eliminating or grouping together certain hashtags. For example, the hashtag "President" refers to one person in the United States, and another in Russia. The top 500 populated cities and their country pair are saved in an offline data-file with their latitude and longitude. Each tweet is clustered to its closest city and country. When comparing a hashtag with a tweet, the closer the associated hashtag is physically with the tweet, the higher the score associated with that tweet. If a hashtag is commonly used at locations far away from where the query tweet is located, then it will receive a lower score and be less likely to be assigned to that tweet.

3.7 Term Frequency – Inverse Document Frequency

Now that all the features have been filtered and created, we need to calculate how important each feature is for each tweet. For the purposes of classification, each word needs to be assigned a value so that it can be placed into a vector and compared with against other vectors. One of the simpler forms of TF-IDF calculation was employed. The first thing was to count the number of times each word was used. If the word was used in 2 documents, it got a count of 2. If the word occurred 3 times in the 2 documents, it still was given a count of 2. This gives us an overview of the word distribution over all the tweets or the Document Frequency of a word. Next, we need to compute the Term Frequency, or how many times a word occurs within a tweet. So if a word occurs 2 times in a tweet, it gets a count of 2. Finally, to compute the TF-IDF for a particular word in a tweet, we take the Term Frequency of a particular word and multiply it by the Log of the number of tweets divided by Document Frequency of the same word.

$$IDF(word) = Log(\frac{NumberOfTweets}{wordCount})$$

$$TF(word,\ Tweet) = Number\ of\ times\ (word)\ occurs\ in\ (Tweet)$$

$$TFIDF(word,\ Tweet) = TF(word, Tweet) * IDF(word)$$

The idea is that if a word occurs across many documents, its weight will be weighted lowed by the Inverse Document Frequency. That way if the word "the" occurs in a tweet more times than "Obama", it won't get a higher score than "Obama" since clearly "Obama" is more important than the word "the".

3.8 Sentiment Analysis

A separate feature that's being used to calculate the hashtag associated with a tweet is the sentiment or emotion associated with the tweet. For example, if a tweet is about something sad or depressing, it doesn't make sense to associate it with a happy hashtag, or the probability of the happy hashtag being associated with the tweet should be lowered.

A method similar to that used in [1] is used where there is a pre-determined database of words and emoticons associated with a set of emotions. A database consisting of 18 emotions was developed: Open, Happy, Alive, Good, Love, Interested, Positive, Strong, Angry, Depressed, Confused, Helpless, Indifferent, Afraid, Hurt, Sad, Lonely, Silly, and None.

Each emotion is associated with a set of words and emoticons. When evaluating the sentiment of a tweet, the words for each emotion are compared with each word in the tweet. If the word exists in the tweet, then the emotion gets a +1 vote. The emotion with the highest vote wins and the score associated with the emotion is stored as the value for that emotion.

The primary reason for using a pre-defined database of emotions and words associated with them is because there is no fixed set of emotions with each tweet. Otherwise a clustering algorithm or something similar could be used to learn words associated with each sentiment. The database consists of at least 15 words with each emotion, with an average of 40 words or emoticons associated.

When a tweet is associated with a sentiment, any hashtag associated with the tweet gets a vote for that sentiment. If a hashtag is used in multiple tweets and if the tweets have different sentiment types, then the sentiment with the highest score is assigned, but the value of the sentiment assigned is subtracted by the lesser value to indicate that the sentiment type is not as strongly associated with the hashtag.

3.9 Hashtag Generation

The goal of this paper is to design an algorithm to assign each tweet with an appropriate hashtag. So far we've only been working with tweets that already contain hashtags. The main reason is that we know for certain that those hashtags go with those tweets. However, this may not be enough as you may still have tweets that would need hashtags that are not covered by the given hashtags, or all of the current hashtags would be inappropriate to assign to some tweets. To get around this, for those tweets that do not have hashtags already associated with them, we generate "guess" hashtags and assign them. The guessed hashtag is the most important word, or highest scoring word from the TF-IDF stage. The reason is that this word is hopefully the most interesting word in the tweet or the most descriptive. As these hashtags are being generated, if a hashtag is generated that has already been generated or already exists, its feature vector is combined with the others to form a single feature vector representing the hashtag. The vectors are combined by taking the average of the values for each word in the vectors and the sentiment values and types are combined as described earlier along with the geographic locations.

3.10 Classification

Given all of the filtered and gathered tweet data, along with the generated hashtags and calculated feature vectors, it's now time to classify and assign hashtags to tweets. We decided to compare the results of three different classifiers, a weighted voting classifier, a nearest neighbor classifier, and a support vector machine classifier. The main criterion was that the classifiers had to be able to perform multi-class classification. The main difference between these classifiers is that the weighted voting classifier and the nearest neighbor can produce more than one hashtag per tweet.

The weighted vector classifier works by taking two vectors, one representing the tweet, the other representing a tentative hashtag. For each word in the feature vectors, if the words are in common (meaning the feature values are not zero), then the average value between the vectors for that word is calculated and added to an overall score. If the vectors do not share the word in common then something more complicated happens. If the tweet vector contains the word but the hashtag vector does not, then 40% of the value of the word in the tweet vector is subtracted from the overall score. If the hashtag vector contains the word and the tweet vector does not then the hashtags vector word value multiplied by one over the length of the hashtag vector is subtracted from the overall score. The idea behind these operations is if the tweet does not contain a word that the hashtag vector does then it should lose more score than if the hashtag vector not containing it. It's more important that the

tweet vector matches with the hashtag vector more than the hashtag vector matches the tweet vector. Also, the hashtag vector is likely to have less zero valued features than the tweet since a hashtag consists of the values of multiple tweets. In the end, if the resulting score is above zero, then the hashtag is assigned to the tweet.

The Distance classifier works by assigning the closest matching hashtags to a tweet. Given a tweet vector and a hashtag vector, the distance between their vectors is calculated. The main problem is most vectors are very close together in high dimensional space. It has been shown that in very high dimensionality, the vectors converge. This phenomenon is also known as the "Curse of Dimensionality" [5]. More important is the fact that most values in the vectors are going to be zero, making the distances come out very close. Initial tests showed that without additional parameters, almost every hashtag will get roughly the same score. The additional constraints were that for the hashtag to be considered, it needed to have more than one example tweet to reference, making more of the features in the hashtag vector non-zero. Another constraint is that in this scenario the sentiment and geographic locations have more weight in pairing hashtags than in the weighted voting. The idea is that we want to have features really wean out hashtags, so by making the sentiment type and geographic location more important, it will make other different hashtags even less likely.

The last classifier tested with is the Support Vector Machine. The Java library Weka was used to train and test the classifier. First the classifier needs to know information about the structure of the vectors, such as which parts are features and where the class is. Then you feed in the hashtag vector information and train the model. Once the training is complete, the model is saved and can be queried on test tweet vectors.

4. EVALUATION

4.1 Feasibility Plan

The method takes advantage of various libraries designed to deal with Twitter data and classification. Given the captured tweets and associated data, they will be filtered, cleaned, and then sent into one of the described classifiers. Clustering and Support Vector Machines have been well studied and have been used in the past to have great effect and performance on classifying data [7, 10]. The data being worked with is primarily text based and falls into the domain of Natural Language Processing (NLP). NLP is a well-studied and has been implemented in many different commercial programs and systems. The challenging part of this work is measuring accuracy and success on a primarily unlabeled dataset.

4.2 Empirical

4.2.1 Metrics

The metrics for determining how well the classification was handled is limited. Proportionally few of the tweets collected came with user-given hashtags. Those hashtags are the only "truth" known. For any other tweet, an assigned hashtag is purely the result of the output of the program and there is no comparison or automatic check that can be made. Rather, a human evaluation of the generated and assigned hashtags to unlabeled data is required and performed. For the labeled Twitter data, the number of correct hashtags and number of misclassified hashtags are kept to evaluate how well the classifiers are doing. Note that the hashtags being used for this numeric evaluation are those given from real users, not the automatically generated hashtags. For example, a particular tweet could have the hashtags ["Weather", "Sunny"] assigned to it from a Twitter user. If the classifier labels that tweet with the hashtag ["Weather"], then it got half of the correct hashtags. If it generated ["Weather", "Obama"] then it got half the correct hashtags, but also generated an incorrect hashtag for the tweet. In a perfect world, the classifiers would only generate the correct hashtags for the appropriate tweets and would not use any hashtags that were not pre-labeled to those tweets. However, due to the nature of words, it may be that for the previous example, "Obama" is actually a correct usage of the hashtag because Obama talked about the weather. Even so, it's difficult to determine that automatically and so it will be counted as incorrect. However, this will be taken into consideration during the human evaluation.

For the human evaluation, each set of hashtags for each of the different configurations of the program will be looked over and have a sanity check. Unfortunately, there are thousands upon thousands of tweets, so an overall evaluation of the output will be given along with a few sample successful and incorrect hashtag assignments.

4.2.2 Results

The results are presented as follows: the classification program was run with or without certain properties such as having spell check enabled and using sentiment analysis, using the geo-locations, or keeping the hashtag word in the tweet for each of the classifiers; Weighted Voting, Distance, and SVM. This shows which features worked best and in what combination for each classifier. For the set of tweets that did not have spell check, there were 663,181 features. The tweets that did have spell check had their feature space reduced to 50,083 features. Both the Weighted Voting and Distance classifiers took on average 3 hours to run a single test. The SVM classifier is discussed next.

4.2.2.1 SVM Classifier

The first thing to note is the lack of any results using the Support Vector Machine. The reason is the SVM failed to generate worthwhile results within a timely manner. The SVM was trained with the same amount of data as the other classifiers over the course of a few days, making it the longest to train and test. After the model was generated and saved, it was evaluated using the same data as it was trained with. In the end, most tweets were assigned a seemingly random hashtag without any intelligence about it. The reason is that the multi-class SVM is only able to output a single class, or hashtag, limiting its usefulness. One reason for the bad performance of the classifier is probably the size of the feature space. As stated previously s, there were over half a million words that the SVM had to classify on, making it too large with not enough examples of each word, to make any intelligent decisions.

4.2.2.2 Distance Classifier

The next set of tests was run using the K-Nearest-Neighbor classifier. For the purposes of this experiment the top 7 hashtags were assigned to a particular tweet. Given a hashtag and tweet pair, the distance between the vectors representing them was calculated and the pairs that had the smallest distances were output. Figure 1 shows the percentage of correct hashtags assigned to a tweet along with the number of incorrectly assigned hashtags. The number of incorrectly assigned hashtags is divided by 100 to make the charts easier to see.

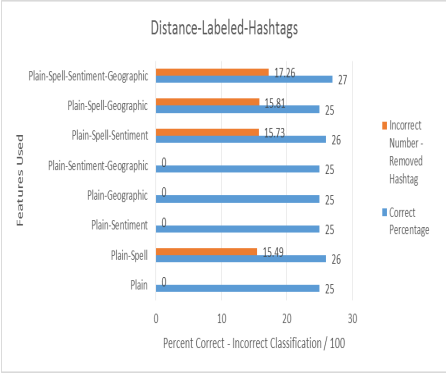

Figure 1: Shows the output for the Distance classification. The percent of correct hashtag assignments is relatively low, on average 25% of the hashtags are correctly labeled. However, many of the tests show on average low incorrect hashtag assignments.

Figure 1 shows the results of using the Distance classifier to assign user defined hashtags to their respective tweets. On average, the number of correct assignments is low, roughly 25%. However, the number of incorrect hashtag assignments is also on average low (compared to the Voted classifier). It appears that the spell checking feature is what's causing the incorrect hashtag assignments. Although, it does appear that by adding spelling, you on average receive a 1% increase in correct hashtag assignments, but at the cost of a large number of incorrect assignments.

The next set of tests is exactly the same except for the hashtags left in the tweets. This was to see how well the classifier does when the "correct answer" is within the tweet.

Figure 2: Shows the output for the Distance classification with the hashtags left in the tweets. The correct percentages are exactly the same as in (Figure 1), but with the number of incorrect hashtags increasing very slightly.

Figure 2 shows the results of the same Distance classifier, except for the hashtags left in the tweets. This set of tests gave the same output for the number of correct assignments. However, when leaving the hashtag in the tweet, the number of incorrect assignments increased very slightly. The reason for the increase in error is not entirely understood. A conjecture is that since the increases are only on those tests using spell check, it's possible that the hashtags are being changed to a different word. By changing the hashtags to a different or totally random word (in the case of very abstract hashtags), it can make the classification of a particular tweet very difficult as it has to now deal with these strange new words. A future change would be not run the spell checker on the hashtags within a tweet or to develop a more sophisticated spell checking system.

4.2.2.3 Voting Classifier

The last set of tests were run using the Voting classifier. The classifier assigned a score to each hashtag and assigned a hashtag to a tweet as long as the final score was above zero.

Figure 3 shows the output from the Weighted Voting algorithm. It shows that most of the time, it correctly re-classified the right hashtags with the right tweets. However, it shows massive amounts of incorrect hashtags being labeled with tweets. It does turn out that many of the incorrectly labeled tweets fall into the category of actually fitting with the tweet. For this particular data-set, most of the tweets are about jobs, so many of the hashtags added to the tweets were job related.

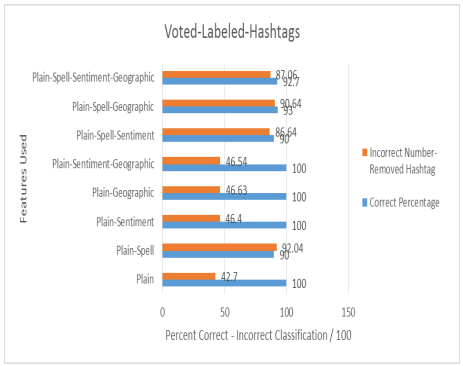

Figure 3: Hashtag Weighted Voting classification of the twitter labeled data. The Y-axis shows the different features used in the classification process. The X-axis shows the percent the classification got correct and the number of incorrect hashtags assigned divided by 100 for ease of viewing.

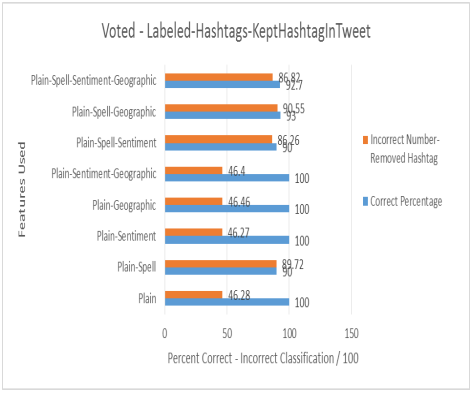

Figure 4: Shows the output for the Weighted Voting classification, exactly the same as in (Figure 3) except the hashtags in the tweets were kept. This is primarily a sanity check; we expect that the correct percentage should be the same or higher, and the incorrect number to be lower.

Figure 4 shows the results of the Weighted Voting classifier on the twitter data except that the hashtags were left in the raw tweet. By leaving the hashtag in the raw tweet, it gives the classification algorithm the feature that it can "cheat on". We expect, and is shown, that the correct percentage is the same or higher, with the incorrect number lower than in the last test (Figure 3). To make comparing the number of incorrect labeling easier, (Figure 5) below shows the comparison with the kept and not kept hashtags.

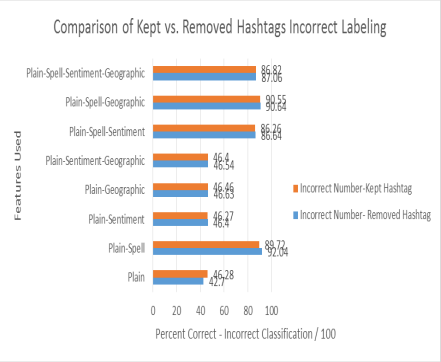

Figure 5: Shows the comparison of the number of incorrect labeling of hashtags in the dataset that kept the hashtags in the tweets compared to the dataset that removed them.

Figure 5 shows that when we keep the original hashtag in the tweet when performing the hashtag discovery, the number of incorrect hashtag labeling slightly decreases. What all these results show is that the voting clustering classifier maintains a high level of correctness, in that it will correctly assign tweets their original hashtags. However, it also assigns a lot of erroneous, extra, or not previously labeled hashtags to tweets, producing the large number of incorrect labeling shown in Figures 3 and 4.

Overall, it can be summarized that the distance classification is more conservative in assigning hashtags, and that the voting classifier is more generous when assigning hashtags.

4.2.2.4 Generated Hashtag Results

Our goal was was to be able to assign hashtags to tweets that did not already have hashtags associated with them. As previously stated, there is no good way to evaluate how well the unlabeled tweet hashtag assignment performed other than by looking at them manually. A list of well assigned tweets (in quotes) and their assigned hashtags (in brackets) are presented below. Note that these tweets did not have any hashtags before running the algorithm.

Successful Labeling and sentiment classification:

- "Good Morning. In my way to church" [CHURCH, JESUS, MORNING, SUNDAY] Sentiment: HAPPY
- "I wish I had a friend I can just go over all the time" [FRIEND, HANG, TRIPS] Sentiment: SAD
- "Come one come all here at my church's Thanksgiving Outreach to the community. @ Downtown Los Angeles" [WELOVELA] Sentiment: NONE
- "What a small world" [DISNEYLAND, DEJAVU, NEIGHBORHOOD] Sentiment: None
- "Great news!" @shayan4congress: ICYMI: Historic nuclear deal reached with Iran in Geneva" [CNN, DEAL, IRANTALKS, MIDEAST, NUCLEAR, TALKS] Sentiment: HAPPY
- "Historical nuclear deal with Iran. peace with Obama. War with Republicans. just one reason why I voted for this man." [REMARKABLE, IRANTALKS, CBC, CNN, IRAN] Sentiment: NONE
- "@CNN: More than 11,000 Syrian children killed in civil war, report says. Horrible" [ASSAD, CHILDREN, HURT, HORRIBLE, WAR, MIDEAST, SYRIA, TALKS] Sentiment: NONE

Bad Labeling:

- "A Kiss on the forehead is One of the Sweetest thing in the world" [ASSAD, GENEVA, SYRIA, MIDEAST] Sentiment: HAPPY
- "I'm at John Keells Computer Services" [HEALTHCARE, JOB, TWEETMYJOBS] Sentiment: NONE
- "Just played a soccer game in twenty degree weather" [GUATEMALA, SOCCER, NHL] Sentiment: NONE

Overall, when the hashtag assignments are correct, they can be very surprising and show that the system works very well. When the classifications are wrong, they're usually very wrong. After going through all the results there are a few trends that we noticed that lead to incorrect classifications. Overall, most tweets were correctly labeled. The ones listed above were among the most interesting because they were labeled with hashtags that are words not present within the tweet itself.

First, it appears that most tweets that include the word "world" usually are given hashtags associated with the Mid-East or Syria. This is because most tweets that have the word "world" in them are associated with the conflicts is Syria and the Nuclear talks in Iran. Another trend is many tweets were assigned the hashtags Healthcare, Job, and Tweet-My-Jobs. After looking at the labeled tweets that have those hashtags, a few things stand out. Either those tweets are extremely short, only having one or two words within them, or they have no words and only have the hashtags in the tweet. What this means is that if you remove the hashtags from the tweet, you have no words or text in the tweet. When you develop of the vector that represents the tweet, all the values are zero, making any distance or score calculation very close to passing. This is reflected in the score and distance calculations as all those hashtags have very low scores (just enough to be accepted), and very low distances (making them in our system, very good hashtags). A way to correct this is to not allow tweets with hashtags to be used as training data as long as it contains text in the tweet after the hashtags are removed, and there are numerous examples. Lastly, for the last bad example shown, two out of the three hashtags make sense. After looking for tweets that had Guatemala, there were two that had talked about soccer. So this means that, while Guatemala does fit the criteria for being about soccer; it's not really appropriate for this tweet.

The tweets presented in the good and bad examples are only a small subset of the good and bad tweets. Overall, the tweets that had incorrect hashtag labeling had similar problems as the ones given. The tweets presented were meant to give an average overview of the types of tweets correctly and incorrectly labeled and overall why the labeling succeeded or failed.

Given all the different combinations of features used when labeling hashtags, it turns out that those that did not use the spell checker turned out better than those that did. The main problem is probably that the spell checker reduces the feature space too much, and probably incorrectly spell checks words, removing or altering important words within a tweet. The feature that seemed most important was the sentiment analysis as it seemed to help prune inappropriate hashtags. At the moment, the geographic location feature did not seem to make much of an impact. The reason is probably that since most of the tweets were filtered to be in English as we used the location of the tweeter, the tweets primarily fell within English speaking countries, primarily the United States of America.

5. CONCLUSION

The goal of this project was to explore and develop a system to discover hashtags for unlabeled tweets. The purpose is to allow the filtering and search of specific tweets based on the hashtags. Different features of tweets were used such as the words, the sentiment of the tweet, and the location of the tweet, to help assign hashtags to them. Overall, the algorithm was successful in that it was able to appropriately label tweets with hashtags that it were not previously associated with the tweet. It was able to generate new hashtags to map to tweets if none of the user-generated tweets applied. However, there were mistakes and incorrect hashtag assignments generated in the algorithm. Given the nature of clustering algorithms, the more data given to the system, the better is will perform. Therefore, in the future, if there is more data given to train the classifier, the there is reason to expect that the system would do better.

6. REFERENCES

[1] D. Davidov, O. Tsur, and A. Rappoport. 2010. Enhanced sentiment learning using Twitter hashtags and smileys. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters* (COLING '10). Association for Computational Linguistics, Stroudsburg, PA, 241-249.

[2] C. Esperança and H. Samet. Experience with SAND/Tcl: a scripting tool for spatial databases. *Journal* of Visual Languages and Computing, 13(2):229–255, Apr. 2002.

[3] N. Gramsky and H. Samet. Seeder finder - identifying additional needles in the Twitter haystack. In Proceedings of the 5th ACM SIGSPATIAL International Workshop on Location-Based Social Networks (LBSN'13), pages 44–53, Orlando, FL, Nov. 2013.

[4] J. Huang, K. M. Thornton, and E. N. Efthimiadis. 2010. Conversational tagging in Twitter. In *Proceedings of the 21st ACM conference on Hypertext and hypermedia* (HT '10). ACM, New York, 173-178.

[5] P. Indyk and R. Motwani. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing* (STOC '98). ACM, New York, 604-613.

[6] A. Jackoway, H. Samet, and J. Sankaranarayanan. Identification of live news events using Twitter. In Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Location-Based Social Networks (LBSN'11), pages 25–32, Chicago, Nov. 2011.

[7] A. K. Jain, M. N. Murty, and P. J. Flynn. 1999. Data clustering: a review. *ACM Comput. Surv.* 31, 3 (September 1999), 264-323.

[8] R. Krikorian, Aug 16, 2013, Platform Engineering , https://blog.twitter.com/2013/new-tweets-per-second-record-and-how

[9] M. D. Lieberman, H. Samet, J. Sankaranarayanan, and J. Sperling. STEWARD: architecture of a spatio-textual search engine. In Proceedings of the 15th ACM International Symposium on Advances in Geographic Information Systems *(GIS'07)*, pages 186–193, Seattle, WA, Nov. 2007.

[10] L. M. Manevitz and M. Yousef. 2002. One-class SVMs for document classification. *J. Mach. Learn. Res.* 2 (March 2002), 139-154.

[11] M. Michelson and S. A. Macskassy. 2010. Discovering users' topics of interest on twitter: a first look. In *Proceedings of the fourth workshop on Analytics for noisy unstructured text data* (AND '10). ACM, New York, 73-80.

[12] Ozdikis, O., Senkul, P., and Oguztuzun, H. (2012). Semantic expansion of hashtags for enhanced event detection in Twitter. In *Proceedings of the 1st International Workshop on Online Social Systems*.

[13] J. H. Parmelee, S. L. Bichard, 2011, Politics and the Twitter Revolution: How Tweets Influence the Relationship between Political Leaders and the Public, Lexington Books.

[14] H. Samet, M. D. Adelfio, B. C. Fruin, M. D. Lieberman, and B. E. Teitler. Porting a web-based mapping application to a smartphone app. In Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems *(GIS'11)*, pages 525–528, Chicago, Nov. 2011.

[15] H. Samet, H. Alborzi, F. Brabec, C. Esperança, G. R. Hjaltason, F. Morgan, and E. Tanin. Use of the SAND spatial browser for digital government applications. Communications of the ACM, 46(1):63–66, Jan. 2003.

[16] H. Samet, A. Rosenfeld, C. A. Shaffer, and R. E. Webber. A geographic information system using quadtrees. Pattern Recognition, 17(6):647–656, November/December 1984.

[17] H. Samet, J. Sankaranarayanan, M. D. Lieberman, M. D. Adelfio, B. C. Fruin, J. M. Lotkowski, D. Panozzo, J. Sperling, and B. E. Teitler. Reading news with maps by exploiting spatial synonyms. Communications of the ACM, 57(10):64–77, Oct. 2014.

[18] H. Samet, B. E. Teitler, M. D. Adelfio, and M. D. Lieberman. Adapting a map query interface for a gesturing touch screen interface. In Proceedings of the 20th International Word Wide Web Conference (Companion Volume), pages 257–260, Hyderabad, India, April 2011.

[19] J. Sankaranarayanan, H. Samet, B. Teitler, M. D. Lieberman, and J. Sperling. TwitterStand: News in tweets. In Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems *(GIS'09)*, pages 42–51, Seattle, WA, Nov. 2009.

[20] C. A. Shaffer, H. Samet, and R. C. Nelson. QUILT: a geographic information system based on quadtrees. International Journal of Geographical Information Systems, 4(2):103–131, April–June 1990. Also University of Maryland Computer Science Technical Report TR–1885.1, July 1987.

[21] B. Teitler, M. D. Lieberman, D. Panozzo, J. Sankaranarayanan, H. Samet, and J. Sperling. NewsStand: A new view on news. In Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems *(GIS'08)*, pages 144–153, Irvine, CA, Nov. 2008.