Topical Clustering of Tweets

Kevin Dela Rosa, Rushin Shah, Bo Lin, Anatole Gershman, Robert Frederking

Language Technologies Institute
Carnegie Mellon University
5000 Forbes Ave. Pittsburgh, PA, USA

{kdelaros, rnshah, bolin, anatoleg, ref}@cs.cmu.edu

ABSTRACT

In the emerging field of micro-blogging and social communication services, users post millions of short messages every day. Keeping track of all the messages posted by your friends and the conversation as a whole can become tedious or even impossible. In this paper, we presented a study on automatically clustering and classifying Twitter messages, also known as "tweets", into different categories, inspired by the approaches taken by news aggregating services like Google News. Our results suggest that the clusters produced by traditional unsupervised methods can often be incoherent from a topical perspective, but utilizing a supervised methodology that utilize the hash-tags as indicators of topics produce surprisingly good results. We also offer a discussion on temporal effects of our methodology and training set size considerations. Lastly, we describe a simple method of finding the most representative tweet in a cluster, and provide an analysis of the results.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *clustering, information filtering, selection process*
I.2.7 [**Artificial Intelligence**]: Natural Language Processing – *text analysis*

General Terms

Algorithms, Experimentation.

Keywords

Social Media, Clustering, Summarization, Microblog Analysis.

1. INTRODUCTION

Recent research efforts in social media analysis and natural language processing have focused on interesting uses of Twitter messages, or "tweets" as they are more colloquially known, and other short socially communicated messages, such as SMS and micro-blogging messages or comments. One interesting problem in tweet analysis is the automatic detection of topics being discussed in tweets. We propose that the hash-tags that appear in tweets can be viewed as approximate indicators of a tweets topic. Furthermore, we propose that standard document clustering and classification techniques from the field of information retrieval can be used to cluster tweets into coarse and fine-grained topics.

In this paper we first discuss past work on tweet and micro-blogging message analysis. Next we formulate our approach to Twitter message topic detection, target topics and describe our data set. Then we describe a set of experiments and results. Next we describe a simple method of summarizing the tweets in a given cluster. Finally we offer a discussion of our results and suggest research future directions.

2. BACKGROUND & RELATED WORK

The analysis of Twitter and various micro-blogging messages is a research area with high and rapidly growing interest within the academic community [1]. Because of the relative freshness of this research area, some research problems have been poorly defined and new problems are being defined every day.

In recent years, researchers have focused on problems such as the summarization and detection of topics for Twitter messages, as well as the mass clustering of tweets. For example, TweetMotif [2] takes an unsupervised approach to message clustering. One issue with this study is that O'Connor does not report metrics on the systems performance, nor does he comment on the generalizability of the approach. We believe that this is partly due to a lack of applicable performance metrics and gold standard labels. Another application of unsupervised methods on Twitter messages was a study by Eisenstein et al. [3], focused on predicting the geo-location of a tweet based on the text in the tweet, which made use of the geo-tagged information in the tweets as the gold standard label for measurement.

Previous research has also exploited the use of supervised methods for topic categorization of short social messages. For example, Ranganath et al. [4] presented a system that detects a speaker's intent to flirt using a spoken corpus of speed-dates; however, their dataset requires human transcription and heavy annotations. Dela Rosa and Ellen [5] also completed a set of experiments on classification of military chat posts, another form of short social messages, with algorithms such as support vector machines, k-nearest neighbors, Rocchio, and Naïve Bayes. However, like the Ranganath et al. study, their approach relies heavily on annotated data, which is usually not available for large-scale micro-blogging messages, like those encountered in Twitter.

In the following sections, we describe how we leverage the largely available hash-tags in the tweets as the gold standard, and describe the algorithms and dataset we use to test our methodology.

3. PROBLEM DEFINITION

In this paper, we experiment with ways of clustering tweets into six predefined topics: News, Sports, Entertainment, Science, Technology, Money, and "Just for Fun". To define our clusters, we leverage popular hash-tags that appear in the tweets for a given topic as a sort of gold standard label used by our different clustering and categorization algorithms. Hash-tags are keywords prefixed with the '#' symbol that can appear anywhere in a tweet, which Twitter users use to categorize their tweets and enabled

SWSM'10, July 28, 2011, Beijing, China.

them to be more easily found in search. The exact hash-tags we used in this study are described in section 4. In our experiments, we look at clusters at both the fine-grained level (raw-hash-tags), and at a coarse-grained level (i.e. our six predefined categories).

This formulation of the problem of the topical clustering of tweets is motivated by the observation that hash-tags are approximate indicators of tweets topics [6]. This study was inspired by an emerging need to cluster user tweets into topics in an analogous way to how Google News perform clustering on news articles. A huge volume of messages are being shared on Twitter every hour, and it is clear that academia and/or industry will have to start to address the issue of information overload, and we feel that this sort of topical clustering of tweets can help in addressing that.

While leveraging the largely available hash-tags found in tweets helps us overcome the issue of un-annotated gold standard labels, there are many other issues that make this problem interesting and difficult. For example, tweets are very short by nature, around 10-15 word tokens on average due to the 140 character limit imposed by Twitter, and full of jargon, misspellings, and abbreviations. This means that researchers must be careful in how they handle the data, as the words in tweets are very sparse, which can be an issue for standard document clustering and classification algorithms. Furthermore, the highly informal nature of the vocabulary found in tweets can make certain vocabulary normalization methods, such as stemming, perform poorly.

In this study, the main research questions we plan to address are as follows:

- **Coarse-grained clustering**: Can we cluster/categorize tweets into our six predefined topics?
- **Fine-grained clustering**: Can we cluster/categorize tweets into the sub-topics (i.e. raw hash-tags) that make up the more general topics?
- **Topic Drift**: How do our language models hold as time passes, and does the change in content in the tweets result in a significant topic drift?
- **Summarization**: Can we find the most representative tweets in a given cluster, in order to summarize the discussions?

4. TWEET CORPUS

To evaluate our topical clustering techniques, we created a data set consisting of tweets collected through Twitter's search API [7] during the second and third week of March 2011. To seed our Twitter message search, we selected a total of 30 popular hash-tags that cover our 6 predefined topics, based on what was trending and in the news prior to collection and queried the Twitter API for a maximum of 400 tweets per hash-tag every hour for two weeks. Table 1 lists the different hash-tags we collected tweets for, and which of the predefined topics they belong to. For the remainder of this paper, our predefined topics will be referred to as coarse tags, and the term fine tag will be used interchangeably with hash-tags.

In total, we collected 1,107,007 tweets contain our target hash-tags (excluding duplicates returned by the Twitter API). Approximately 19% of the Twitter messages were "re-tweets" (RTs), or messages that user's shared with their followers which originated from another user. The average length of a tweet in our corpus was 15.22 word tokens and 105.27 characters long. The total number of 16,847,496 word tokens appeared in our corpus, from 1,555,101 unique word types. Furthermore, on average there were 1.455 hash-tags per tweet, and 65.72% of the tweets had at least one hash-tag, excluding the hash-tags listed in Table 1.

Table 1. Hash-tags and predefined target cluster topics.

Topic (Coarse Level)	Hash-Tags(Fine Level)
News [#news]	#Japan, #wiunion, #obama, #libya, #gop
Sports [#sports]	#knicks, #heat, #nfl, #nba, #nhl
Science & Technology [#science]	#xoom, #ipad2, #google, #nasa, #facebook
Entertainment [# entertainment]	#Bieber, #sheen, #Oscars, #Radiohead, #ladygaga
Money / Business [#money]	#oil, #irs, #tax, #gas, #gold
Just for Fun [#justforfun]	#twitterpetpeeve, #marchwish, #blackpeoplemovies, #tigerblood, #winning

4.1 Normalization & Tokenization

In order to facilitate our experiments, a few of normalization steps had to be performed on the vocabulary of the tweet corpus to reduce the feature space. We experimented with four variations of our vocabulary, all of which were calculated on the full training portion of our corpus. The following are the vocabularies we created:

- **V0**: Words are whitespace tokenized, and put in lowercase.
- **V1**: Words are whitespace tokenized, put into lowercase, and rare terms removed (terms with document frequency < 5 were eliminated).
- **V2**: Words are whitespace tokenized, put into lowercase, all non-alphanumeric characters (aside from the following special characters used in Twitter: _@#-) and rare terms removed.
- **V3**: Words are whitespace tokenized, put into lowercase, URLs and usernames mapped to two special classes respectively, and all non-alphanumeric characters (aside from the following special characters used in Twitter: _@#-) and rare terms removed.

In total the number of resulting word types in each of the vocabularies are 685753, 77992, 42181, and 39091 for **V0**, **V1**, **V2**, and **V3** respectively. As you can see, the largest drop in vocabulary size is between **V0** and **V1**, yet, as you will see in our experiments, we were able to achieve comparable results with this greatly reduced feature set.

5. UNSUPERVISED CLUSTERING

Table 2. Clustering results of unsupervised approaches

LDA	Purity	Pairwise F-Score
Fine	0.131	0.048
Coarse	0.304	0.143
K-means	**Purity**	**Pairwise F-Score**
Fine	0.285	0.058
Coarse	0.412	0.143

To get an intuition on how hard this problem is, we first ran some unsupervised clustering algorithms on a portion of our dataset. In particular, we used the generative modeling approach of Latent Dirichlet Allocation (LDA) [8] and the vector space based K-means clustering algorithm [9]. For LDA, we ran two experiments, with 30 and 6 topics respectively. Ideally, if tweets tended to cluster together along topic lines, one would expect to see each LDA topic correspond to an actual hash-tag based topic. As we shall see below, this is not the case in reality. For K-means, we used vocabulary **V1** and used TF-IDF weighting of token features. We measure the quality of the clusters produced by these algorithms by using hash-tags to provide a gold standard clustering, and along the following metrics commonly used to evaluate clustering results [9]: Purity & Pairwise F1-Score.

For Purity, each gold standard cluster is first assigned with the most frequent output labels among the members in this cluster, and we then compute the average accuracy of such assignments over all gold standard clusters. Pairwise F1-Score is based on the F1 score on how many pairs of messages are clustered correctly. To calculate this F1 score, we first define true positive (TP) pairs as the pairs of similar messages (i.e. messages in a same gold standard clusters) which are clustered together in the output. False positive (FP), true negative (TN) and false negative (FN) pairs are similarly defined. With these definitions, precision (P), recall (R) and pairwise F1 scores are defined as:

$$P = \frac{TP}{TP + FP} \quad R = \frac{TP}{TP + FN} \quad F1 = \frac{2PR}{P + R}$$

The results for LDA and K-means algorithms are shown in Table 2. For both metrics, the values can be in the interval [0, 1]; where the higher values are better.

We find that although K-means does better than LDA, overall, both methods produce clusters of very poor quality (with respect to our topical hash-tags). This suggests that tweets do not tend to naturally cluster together along topic-based lines, and the problem of topic-based tweet clustering is not inherently easy. Indeed, an analysis of the selected top words for each LDA topic, as shown in Table 3, shows that the topics discovered by LDA are quite different than the semantic topics that we wish to discriminate. In the examples in Table 3, though topic #1 and #3 carry a valid hash-tag based semantic meaning, many others do not. For example, topic #2 is more or less about video / photo uploading services on the web, topic #6 mainly groups together Spanish tweets and topic #11 is just a collection of numbers.

These results show that unsupervised approaches, while convenient, would not present an effective solution to the problem of topical clustering for tweets. Therefore, we turn to the use of supervised approaches. These are covered in the next section.

6. SUPERVISED CLUSTERING

To determine the effectiveness of supervised algorithms on the task of clustering tweets by topics, we performed a range of experiments, on both the fine and coarse-grained levels. In addition to a basic setup, described in Section 6.1, we also run a set of experiments to test the effects of various vocabulary normalization and expansion, the temporal effects on topic drift in tweets, and the performance of classifiers trained on different subsections of the training data. These experiments and their results are described in more detail in the following subsections, and are designed to gain a better understanding of the various aspects and issues involved with clustering tweets by topic.

In each of experiments described in this section, we use the well known Rocchio classifier, which is detailed [9], with simple bag of words features using our different vocabularies. Rocchio was selected because of its broad use in document classification, relative quickness to train, and its ability to handle feature sparsity more robustly without modification than other models, such as Support Vector Machines and K-Nearest Neighbors.

The metrics used in our evaluation are standard to the field of information retrieval: precision, recall, and F1 score. Precision is the proportion of returned tweets that are targets, recall is the proportion of target tweets returned, and the F1 measure is an evaluation metric that combines recall and precision. For some of the experiments, we provide the macro-averaged and micro-averaged values of each of our performance measures. The micro-averaged calculations give equal weight to each tweet, while the macro-averaged values give equal weight to each topic/hash-tag. More details on these metrics can be found in [9].

Table 3. Top words for each topic produced by LDA

Topic	Top Words
1	Egypt Bahrain twitter iphone4 web …
2	Twitpic photo video yfrog tumblr …
3	Nuclear fukushima plant earthquake …
4	Winning charliesheen tsunami prayforjapan …
5	Facebook iphone visit free gaga lady …
6	De en la el los para del una esta …
…	
11	1 3 2 march 2011 11 7 million minutes …
12	Sports nba nfl hockey tspn games …
13	Lol winning charliesheen tonight gonna …
…	

6.1 Basic Experiment

For our basic experimental setting, we trained classifiers for our hash-tags as well as coarse tags. We trained on a subset of 400,000 tweets, and tested on a subset of 50,000 tweets from our corpus, and made sure that the test set had the same class distribution as the training set. Table 4 shows Precision, Recall and F1 scores for each hash-tag and coarse tag using vocabulary **V1**. We also performed this experiment using all four different vocabularies **V0**, **V1**, **V2** and **V3** that were described in a previous section. The results for each of these cases, in terms of micro-averaged and macro-averaged Precision, Recall and F1 are shown in Table 5.

Overall, we find that we are able to achieve surprisingly good precision, recall and F1 scores on the fine-grained hash-tag classification, considering the sparseness of Twitter-style data. Moreover, we achieve even better scores for the coarse tags. This confirms our intuition that classifying tweets according to their broad genres is somewhat easier than identifying their specific topic. Amongst the coarse tags, we achieve impressive performance for all the categories except #justforfun. This is not surprising, considering the whimsical nature of tweets in this category; indeed, the task of accurately detecting humor in text is far from solved. For the fine-grained setting, we find more variance in performance for different tags compared to the coarse-grained setting. Our best precision scores are in the region of 0.7 for tags such as #libya, while our best recall scores are around 0.83 for tags such as #tax. While we achieve good F1 scores (> 0.5) for a majority of the tags, certain tags such as #blackpeoplemovies or #twitterpetpeeve do not perform well. This is again an indication of the difficulty of capturing the essence of more whimsical topics compared to more easily definable topics such as news or sports.

We also find that normalizing the vocabulary by removing all words occurring in less than five tweets (i.e. using **V1** instead of **V0)** does not affect classifier performance appreciably, and yields a significant reduction in the dimensionality of the feature space which leads to much faster training and testing. Therefore, we use **V1** for the rest of our experiments. However, classifier performance begins to degrade with too much normalization, as seen with the numbers obtained using **V3**. The **V3** results also suggest that collapsing all users and URLs into the same vocabulary type leads to the loss of valuable contextual information.

6.2 URL Based Text Expansion

We observed that a large fraction of the tweets in our dataset contain embedded URLs. We hypothesized that retrieving the text from documents pointed to by these URLs might alleviate feature sparsity inherent in short tweets and might improve classifier performance. To test this hypothesis, we took a subset of 45,000 tweets from our original corpus, with the same distribution of hash-tags as the original, and augmented each tweet in this subset by retrieving the text from every URL mentioned in that tweet. We then trained a model on 40,000 of these tweets and tested on the other 5,000. The performance for both cases is shown in Table 6. For this subset of the tweet corpus, 39.88% of the tweets contained URLs, for an average of 0.501 URLs per tweet, and there was an average of 721.51 words on the web pages pointed to by the URLs.

Surprisingly, we find that the inclusion of URL-based text actually hurts the performance of our classifiers, in terms of precision as well as recall. This is quite counter-intuitive; one would expect performance to improve with denser text, but the reverse holds true in this case. Upon closer examination of the text

Table 4. Micro-averaged Precision, Recall and F$_1$ for hash-tags and coarse tags, using vocabulary V1

Fine Tag	**Precision**	**Recall**	**F$_1$**
#bieber	0.682	0.754	0.717
#blackpeoplemovies	0.093	0.107	0.100
#facebook	0.521	0.485	0.502
#gas	0.416	0.418	0.417
#gold	0.672	0.470	0.553
#google	0.739	0.447	0.557
#gop	0.698	0.664	0.681
#heat	0.343	0.632	0.445
#ipad2	0.721	0.524	0.607
#irs	0.915	0.936	0.926
#japan	0.622	0.600	0.610
#knicks	0.295	0.542	0.382
#ladygaga	0.733	0.539	0.622
#libya	0.765	0.694	0.728
#marchwish	0.063	0.517	0.112
#nasa	0.761	0.452	0.567
#nba	0.829	0.577	0.680
#nfl	0.789	0.482	0.599
#nhl	0.825	0.635	0.718
#obama	0.559	0.363	0.440
#oil	0.488	0.413	0.447
#oscars	0.088	0.392	0.143
#radiohead	0.326	0.264	0.292
#sheen	0.176	0.439	0.251
#tax	0.697	0.836	0.760
#tigerblood	0.618	0.437	0.512
#twitterpetpeeve	0.022	0.763	0.043
#winning	0.363	0.503	0.421
#wiunion	0.556	0.660	0.604
#xoom	0.421	0.574	0.486
Micro-average	**0.551**	**0.546**	**0.549**
Macro-average	**0.526**	**0.537**	**0.497**
Coarse Tag	**Precision**	**Recall**	**F$_1$**
#entertainment	0.656	0.509	0.573
#fun	0.527	0.703	0.603
#money	0.609	0.605	0.607
#news	0.767	0.762	0.765
#science	0.721	0.577	0.641
#sports	0.754	0.717	0.735
Micro-average	**0.688**	**0.681**	**0.685**
Macro-average	**0.672**	**0.646**	**0.654**

Table 5. Micro-averaged and Macro-averaged Precision, Recall and F1 scores using different vocabularies

Fine	Micro Prec	Micro Recl	Micro F_1	Macro Prec	Macro Recl	Macro F_1
V0	0.555	0.550	0.552	0.528	0.540	0.500
V1	0.551	0.546	0.549	0.526	0.537	0.497
V2	0.541	0.533	0.537	0.528	0.530	0.492
V3	0.532	0.528	0.530	0.523	0.521	0.486
Coarse	**Micro Prec**	**Micro Recl**	**Micro F_1**	**Macro Prec**	**Macro Recl**	**Macro F_1**
V0	0.688	0.683	0.686	0.672	0.647	0.655
V1	0.688	0.681	0.685	0.672	0.646	0.654
V2	0.683	0.673	0.678	0.666	0.637	0.647
V3	0.674	0.669	0.672	0.666	0.638	0.645

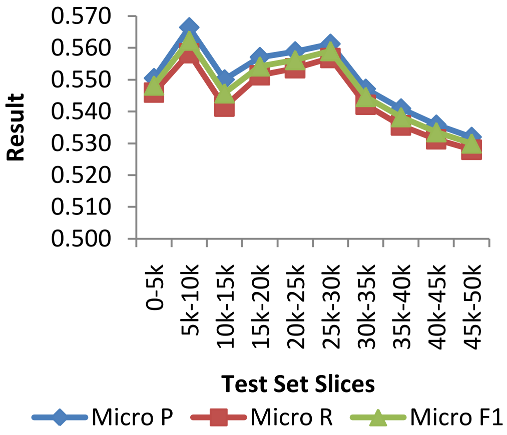


Figure 1. Classifier Performance on temporal slices of our test set

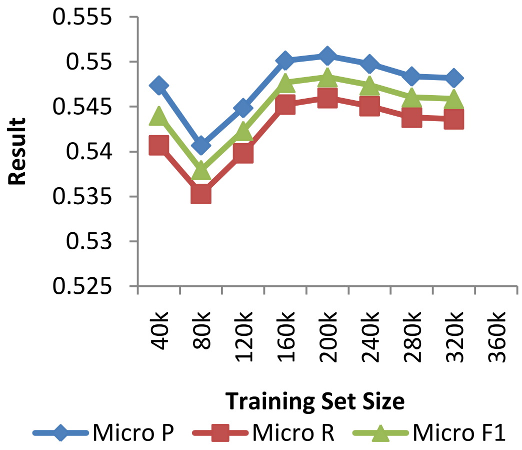


Figure 2. Effect on classifier performance with varying training set sizes

Table 6. Micro-averaged and Macro-averaged scores with and without URL-based text expansion

	Coarse		Fine	
	w/o URL	with URL	w/o URL	with URL
Micro P	0.700	0.665	0.559	0.463
Micro R	0.689	0.657	0.550	0.458
Micro F_1	0.695	0.661	0.554	0.460
Macro P	0.659	0.672	0.530	0.490
Macro R	0.636	0.605	0.513	0.463
Macro F_1	0.639	0.622	0.491	0.401

retrieved from in-tweet URLs, we find that the majority of such URLs happen to be off-topic, and a lot of them are simply spam. As a result, they lead to a less discriminative trained model, and similarly, more unfocused test set tweets. This is an interesting reflection on the quality of embedded URLs currently prevalent in tweets.

6.3 Time Series Analysis of Classifier Performance

We examined how classifier accuracy changes as test set tweets either become more recent or move further away temporally from our training set tweets, to get some insight on the question of how temporally close test set tweets have to be to the training set for supervised models and to identify their topic with a high degree of accuracy. To answer this question, we divided our test set of 50,000 tweets into ten temporally ordered slices of 5000 tweets. We made sure that the slices retained the same class distribution as the original test set. Figure 1 shows a trend line of our classifier's performance across each of these ten slices.

We find that for the first slice (i.e. tweets 0 – 5k ; the test set tweets that are temporally closest to the training set), the precision and recall both tend to be highest, but these scores steadily decrease as we go to the 2nd, 3rd slice and so on. By the 10th slice (i.e. tweets 45k – 50k; the 5000 tweets that are furthest from the training set), we can see significant performance degradation. This confirms our initial assumption, and provides evidence of the topic drift phenomenon in Twitter-style data: the meaning conveyed by a particular hash-tag subtly changes over time.

6.4 Effect of Varying Training Set Size

In addition to examining classifier performance across different temporal splits of our test set, we also observed the effect that the size of our training data has on the performance of our classifiers on a fixed test set. In particular, we analyzed whether a smaller but more temporally focused training set might produce better results than a larger training set accumulated over more time. We used our initial training set of 400,000 tweets to form ten different training sets with increasing size in multiples of 40000, from 40000, 80000, and so on up to 400000. We made sure that each training set contained tweets closest to the test set tweets, since in this particular experiment we do not want topic drift to affect our results. We trained different classifiers using each of these progressively larger training sets, and evaluated them on a common test set of 5000 tweets (the first test set slice, tweets 0 – 5k, from the previous experiment). Figure 2 shows the performances of each of these classifiers.

We see that classifier performance increases with an increase in training set size up to a certain point (200k training instances), but then levels off and even degrades slightly. This suggests that a large training set that stretches over too large a time period can actually be worse in some cases than a smaller but more recently focused training set.

7. CLUSTER SUMMARIZATION

After receiving a cluster of tweets, one natural extension would be to automatically summarize the topics & stories being discussed within the cluster. We propose that finding the most representative tweet or top few tweets in the collection is an effective way of summarizing a cluster, analogous to the way that Google News displays the most relevant news story in a cluster of online news articles. Clusters may contain hundreds or thousands of tweets, and the task of extracting the top few tweets that give an accurate representation of the cluster is not only a significant advantage in many applications, but also an interesting theoretical problem since conceptually we have the dual constraints of making sure that not only are the summary tweets as representative as possible, but also as diverse as possible from each other. Another significant obstacle is that unlike our clustering experiments where we used hash-tags as gold standard class labels, there is no readily available gold standard to measure summarization performance for our tweet corpus, which we address by the use of crowd sourcing for relevance judgments.

In this section, we first describe a simple method for finding the most representative tweets of a cluster. Then we describe a small experiment designed to test our methodology. Next we describe the use of Amazon Mechanical Turk, a popular crowd sourcing platform, for gathering relevance judgments, and finally show the performance of our methodology using standard information retrieval metrics.

7.1 Methodology

In order to find most representative tweets of some collection or cluster of tweets on some specific subject we propose to use a simple and straightforward algorithm based on a document novelty selection technique developed Li et al. for use on the Web Track at TREC 2010 [10].

The algorithm we used to select the top *N* most representative tweets from a given cluster is as follows:

- Construct a centroid, *V*, for the cluster of tweets, *C*
- Initialize an empty list for the selected tweets, *T*
- Sort all tweets in *C* according to their TF-IDF similarity with *V*, where the highest ranked ones are the ones that are most similar
- Loop *N* times
    - Pick the highest ranked tweet, *t*, from *C*, whose TF-IDF similarity against all tweets in *T* is below some threshold *k*
    - Add *t* to *T*, and remove *t* from *C*

Note that in our experiments *k* was set empirically (in the range of 0.4-0.5), to find summaries that seem to be the best in terms of representativeness and diversity.

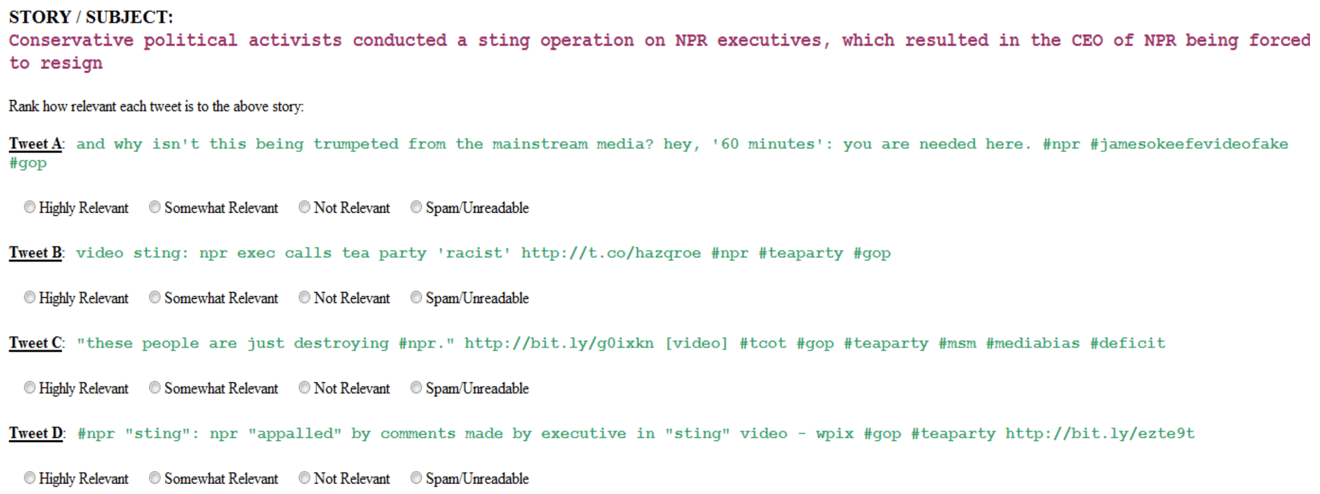
7.2 Experimental Setup

For our experiments, we were interested in finding the most representative tweets for some of the stories/sub-topics we saw emerging from our hash-tag based topical clusters, as we saw this being more interesting and useful than finding the most representative tweets for the topic overall. For example, it would probably be difficult to find a coherent and concise summary of the #obama topical cluster, but it would definitely be easier and more useful to have summarizations for various stories involving #obama that we found in the cluster, such as his reaction to the Japanese tsunami and his rhetoric leading up to the Libya war.

Since we were not interested in the task of cluster decomposition into stories, we manually selected 15 stories based on our corpus, and manually created (with the aid of some unsupervised clustering algorithms like K-means) 15 clusters of tweets, one for each story. On average there were 410.27 tweets in each of the story clusters we constructed. We looked at the following stories:

1. Musical artist Lady Gaga releases a single and video for 'Born This Way', a song from her upcoming album.
2. There are rumors that Google will get more aggressive in entering the social networking market that Facebook currently dominates, and possibly launch a new service.
3. Conservative political activists conducted a sting operation on NPR executives, which resulted in the CEO of NPR being forced to resign.
4. Internal divisions arise within the GOP and Tea Party after the passing of a bill (and during the efforts to pass the bill) containing spending cuts.
5. Wisconsin GOP overcame procedural hurdles caused by Democrats to pass a bill that is widely considered to be anti-union.
6. The Miami Heat suffered a tough loss to the Chicago Bulls (basketball). There are rumors that Miami players were actually crying in the locker room after the game.
7. Social media outlets like Twitter and Facebook were vital modes of communication and donation collection during the Japanese earthquakes and tsunamis.
8. The Los Angeles Lakers beat the San Antonio Spurs on March 6, 2011 (basketball).
9. President Obama plans to appoint Gary Locke to be the new US ambassador to China.
10. President Obama made a statement of support and offers help to Japan in the wake of the earthquakes and tsunamis.
11. President Obama made a statement on a possible U.S. and NATO response to the situation in Libya.
12. In March 2011, Warner Brothers fired Charlie Sheen from the TV show 'Two and a Half Men'.
13. Charlie Sheen released a disturbing video on ustream.tv after getting fired from the popular TV show 'Two and a half men'.
14. Warner Brothers started a movie rental service on Facebook.
15. Mark Zuckerberg recently got a new dog, and was also briefly featured as an action figure by some company.

After constructing this story list and tweet clusters, we ran the algorithm described in Section 7.1 to find the top 10 most



STORY / SUBJECT:
Conservative political activists conducted a sting operation on NPR executives, which resulted in the CEO of NPR being forced to resign

Rank how relevant each tweet is to the above story:

Tweet A: and why isn't this being trumpeted from the mainstream media? hey, '60 minutes': you are needed here. #npr #jamesokeefevideofake #gop

○ Highly Relevant ○ Somewhat Relevant ○ Not Relevant ○ Spam/Unreadable

Tweet B: video sting: npr exec calls tea party 'racist' http://t.co/hazqroe #npr #teaparty #gop

○ Highly Relevant ○ Somewhat Relevant ○ Not Relevant ○ Spam/Unreadable

Tweet C: "these people are just destroying #npr." http://bit.ly/g0ixkn [video] #tcot #gop #teaparty #msm #mediabias #deficit

○ Highly Relevant ○ Somewhat Relevant ○ Not Relevant ○ Spam/Unreadable

Tweet D: #npr "sting": npr "appalled" by comments made by executive in "sting" video - wpix #gop #teaparty http://bit.ly/ezte9t

○ Highly Relevant ○ Somewhat Relevant ○ Not Relevant ○ Spam/Unreadable

Figure 3. Example Amazon Mechanical Turk HIT used for getting relevance judgments of tweets for a given story

representative tweets of each story cluster. The following sections describe how we evaluated the results of this experiment.

7.3 Relevance Judgment

The first step in evaluating our results was assigning relevance judgments to the tweets with respect to a story. Our relevance scores for each tweet were as follows:

- **Highly Relevant (HRel)**: The tweet is either summarizing the story or directly referencing the story / event as it is happening
- **Relevant (Rel)**: The tweet references portions of the story but does not convey the entire message of the story
- **Not Relevant / Spam**: The tweet contains some keywords from story but shares little content that on the story, or is spam, unreadable or in a foreign language.

Since this task of assigning judgments is too time consuming for an individual to do by themselves (and error prone if done by a single individual), we opted to have workers on Amazon Mechanical Turk assign judgments. We posted a series of HITs (Amazon Mechanical Turk tasks), where workers assigned scores to the relevance of a tweet to a story using radio buttons, an example of which is shown in Figure 3 (instructions/guidelines given to the workers are not shown in the figure).

To guard against inconsistencies in ranks, we had 3 different workers judge every tweet, and took the majority relevance score; if no majority was present, the tweet was thrown out of consideration during evaluation. Also, for usability reasons, we showed not relevant and spam as different options, but in reality their scores equivalent in the evaluation metrics. In addition to judging our top 10 tweets per cluster (as determined by the algorithm in Section 7.1), we also selected 10 random tweets from each cluster for use as a comparison in our evaluation.

7.4 Results

To evaluate our results, we used the relevance judgments described in Section 7.3, and calculated some standard IR metrics, such as precision and normalized discounted cumulative gain (NDCG). For the NDCG calculations, we assigned a score of 3 to highly relevant tweets, 1 to relevant tweets, and 0 to non-relevant and spam, in a similar fashion to evaluations carried out at TREC. Table 7 summarizes our performance and compares it to a random tweet selection baseline. As you can see our relevant tweet algorithms is effective, particularly closer to the top, clearly performs significantly better than random, and has decent NDCG@R and Precision numbers.

8. DISCUSSION

We find a number of interesting observations based on our experiments:

- We achieve substantially better scores for coarse tags than for fine-grained hash-tags. In other words, classifying tweets according to their broad genres is easier than identifying their specific topic.
- Although the tweets do not tend to form natural topic-based clusters, as evidenced by the performance of methods such as K-means and LDA, the use of supervised models and training data significantly improves performance for the task of topical clustering.
- The use of relatively simple normalization rules such as removing all words occurring in less than five tweets does not affect classifier performance appreciably, but

Table 7. Performance of Relevant Tweet Selection Algorithm & Random on Cluster Summarization

Precision Metric *HRel is relevant*	**Tweet Selection Algorithm**	**Random Tweets**
P@1	0.4666	0.1333
P@5	0.4800	0.1733
P@10	0.4133	0.2066
Precision Metric *HRel/Rel are relevant*	**Algorithm**	**Random**
P@1	0.8000	0.4000
P@5	0.8533	0.5200
P@10	0.8200	0.4733
NDCG@R Metric	**Algorithm**	**Random**
NDCG@5	0.6911	0.4658
NDCG@10	0.8219	0.6312

yields significant reduction in the dimensionality of the feature space.

- The use of URL-based retrieved text, which should theoretically improve performance by taking the problem closer to standard document classification, actually hurts performance, due to the vast quantity of off-topic documents referenced by those URLs.
- Our classifiers perform well on test data that is temporally close to the training set, and show a steady drop-off with time. This suggests that for online applications such as real-time news aggregation and clustering, one must keep updating one's training data, and use light-weight methods that can be trained quickly, such as Rocchio, as opposed to more computationally expensive methods such as SVMs.
- Similarly, we also found that training sets that got too big and too stretched out across time actually performed worse than smaller, more focused training sets. This is another useful observation for the field of real-time news gathering using social media services such as Twitter.
- Simple centroid and TF-IDF similarity based rankings can be used to find the most representative tweets, which seem to be a useful form of tweet cluster summarization.

9. CLOSING REMARKS

Twitter research is an exciting area of current research, considering the novelty of tweet-style data that are noisy, short, and unlabeled. We collected a large corpus of tweets with temporal information and fine-grained as well as coarse-grained topic labels in the form of hash-tags, which we are happy to make available for academic and research purposes upon request.

We proposed a novel approach to addressing the lack of Twitter annotation for topical clustering by using hash-tags. Our study of topic clustering is inspired by systems like Google News and past work on document categorization and clustering. We found that unsupervised methods tend to classify tweets based on language similarity rather than topical coherence. However, standard supervised methods actually work well on this short and noisy data. Unsurprisingly, coarse-grained clustering of tweets turns out to be easier than fine-grained clustering. We also found that normalization of vocabulary is an effective technique for dimensionality reduction. An interesting result we came across was that augmenting tweets with text retrieved from embedded URLs actually hurt classification accuracy. Furthermore, we found evidence of topic drift if test sets are too far displaced in time from the training set or if training sets are too large and spread out across too much time; this is something to keep in mind for future researchers and developers working with this kind of data.

Lastly, we explored the use of a simple document selection technique applied to finding the most representative tweets in a given cluster, and found it to be surprisingly effective.

We believe that there is a lot of scope for interesting research directions people can take with respect to this problem, such as generalizing our research to perform topic-based clustering in an online setting, automatically detecting hash-tags that correspond to relevant and interesting topics and incorporating features that leverage the graph structure of Twitter to improve classification. For future work, we plan to apply sentiment analysis to better understand the clusters discovered by our models, and to explore other applications of short text and social media clusters.

10. REFERENCES

[1] Ellen, J. 2011. All about microtext: A working definition and a survey of current microtext research within artificial intelligence and natural language processing. In *Proceedings of the Third International Conference on Agents and Artificial Intelligence*.

[2] Eisenstein, J., O'Connor, B., Smith, N.A., and Xing, E.P. 2010. A Latent Variable Model for Geographic Lexical Variation. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*.

[3] O'Connor B., Krieger, M., and Ahn, D. 2010. TweetMotif: Exploratory Search and Topic Summarization for Twitter. In *Proceedings of Fourth International AAAI Conference on Weblogs and Social Media*.

[4] Ranganath R., Jurafsky, D., and McFarland. D. 2009. It's Not You, it's Me: Detecting Flirting and its Misperception in Speed-Dates. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*.

[5] Dela Rosa, K., and Ellen, J. 2009. Text classification methodologies applied to micro-text in military chat. In *Proceedings of the International Conference on Machine Learning and Applications*.

[6] Dann, S. 2010. Twitter Content Classification. *First Monday* (Online). 15, 12.

[7] Twitter API. http://dev.twitter.com/

[8] Blei D.M., Ng, A.Y., and Jordan, M.I. 2003. Latent Dirichlet Allocation. *J. Mach. Learn. Res*. 3 (Jan. 2003) 993-1022

[9] Manning, C.D., Raghavan, P., and Schütze, H. 2008. Introduction to Information Retrieval, Cambridge University Press.

[10] Li, Z.C., Fang, Q., Chen, F., Xing, Q.L., Zhu, T., Zhang, M., Jin, Y.J., and Ma, S.P.THUIR at TREC 2010 Web Track: Revisiting the Use of Anchor Text and Finding Novel Results for Diversification. 2010. In *Proceedings of the Ninth Text Retrieval Conference*.