Comparing Twitter Summarization Algorithms for Multiple Post Summaries

David Inouye* and Jugal K. Kalita+

*School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332 USA

+Department of Computer Science
University of Colorado
Colorado Springs, CO 80918 USA

dinouye3@gatech.edu, jkalita@uccs.edu

***Abstract*—Due to the sheer volume of text generated by a microblog site like Twitter, it is often difficult to fully understand what is being said about various topics. In an attempt to understand microblogs better, this paper compares algorithms for extractive summarization of microblog posts. We present two algorithms that produce summaries by selecting several posts from a given set. We evaluate the generated summaries by comparing them to both manually produced summaries and summaries produced by several leading traditional summarization systems. In order to shed light on the special nature of Twitter posts, we include extensive analysis of our results, some of which are unexpected.**

I. Introduction

Twitter[1], the microblogging site started in 2006, has become a social phenomenon. In February 2011, Twitter had 200 million registered users[2]. There were a total of 25 billion tweets in all of 2010[3]. While a majority of posts are conversational or not particularly meaningful, about 3.6% of the posts concern topics of mainstream news[4].

To help people who read Twitter posts or tweets, Twitter provides two interesting features: an API that allows users to search for posts that contain a topic phrase and a short list of popular topics called *Trending Topics*. A user can perform a search for a topic and retrieve a list of the most recent posts that contain the topic phrase. The difficulty in interpreting the results is that the returned posts are only sorted by recency, not relevancy. Therefore, the user is forced to manually read through the posts in order to understand what users are primarily saying about a particular topic. The motivation of the summarizer is to automate this process.

In this paper, we discuss ongoing effort to create automatic summaries of Twitter trending topics. In our recent prior work [1]–[4], we have discussed algorithms that can be used to pick the single post that is representative of or is the summary of a number of Twitter posts. Since the posts returned by the Twitter API for a specified topic likely represent several sub-topics or themes, it may be more appropriate to produce summaries that encompass the multiple themes rather than just having one post describe the whole topic. For this reason, this paper extends the work significantly to create summaries that contain multiple posts. We compare our multiple post summaries with ones produced by leading traditional summarizers.

II. Related Work

Summarizing microblogs can be viewed as an instance of the more general problem of automated text summarization, which is the problem of automatically generating a condensed version of the most important content from one or more documents. A number of algorithms have been developed for various aspects of document summarization during recent years. Notable algorithms include SumBasic [5] and the centroid algorithm [6]. SumBasic's underlying premise is that words that occur more frequently across documents have a higher probability of being selected for human created multi-document summaries than words that occur less frequently. The centroid algorithm takes into consideration a centrality measure of a sentence in relation to the overall topic of the document cluster or in relation to a document in the case of single document summarization. The LexRank algorithm [7] for computing the relative importance of sentences or other textual units in a document (or a set of documents) creates an adjacency matrix among the textual units and then computes the stationary distribution considering it to be a Markov chain. The TextRank algorithm [8] is also a graph-based approach that finds the most highly ranked sentences (or keywords) in a document using the PageRank algorithm [9].

In most cases, text summarization is performed for the purposes of saving users time by reducing the amount of content

[1] http://www.twitter.com
[2] http://www.bbc.co.uk/news/business-12889048
[3] http://blog.twitter.com/2010/12/hindsight2010-top-trends-on-twitter.html
[4] http://www.pearanalytics.com/blog/wp-content/uploads/2010/05/Twitter-Study-August-2009.pdf

to read. However, text summarization has also been performed for purposes such as reducing the number of features required for classifying (e.g. [10]) or clustering (e.g. [11]) documents. Following another line of approach, early work by Kalita et al. generated textual summaries of database query results [12]–[14]. Instead of presenting a table of data rows as the response to a database query, they generated textual summaries from predominant patterns found within the data table.

In the context of the Web, multi-document summarization is useful in combining information from multiple sources. Information may have to be extracted from many different articles and pieced together to form a comprehensive and coherent summary. One major difference between single document summarization and multi-document summarization is the potential redundancy that comes from using many source texts. One solution may involve clustering the important sentences picked out from the various source texts and using only a representative sentence from each cluster. For example, McKeown et al. first cluster the text units and then choose the representative units from the clusters to include in the final summary [15]. Dhillon models a document collection as a bipartite graph consisting of words and documents and uses a spectral co-clustering algorithm to obtain excellent results [16].

Finally, in the context of multi-document summarization, it is appropriate to mention MEAD [17], a flexible platform for multi-document multi-lingual publicly-available summarization. MEAD implements multiple summarization algorithms as well as provides metrics for evaluating multi-document summaries.

III. Problem Description

A Twitter post or tweet is at most 140 characters long and in this study we only consider English posts. Because a post is informal, it often has colloquial syntax, non-standard orthography or non-standard spelling, and it frequently lacks any punctuation.

The problem considered in this paper can be defined as follows:

> *Given a topic keyword or phrase T and the desired length k for the summary, output a set of representative posts S with a cardinality of k such that 1) $\forall s \in S$, T is in the text of s, and 2) $\forall s_i, \forall s_j \in S, s_i \not\sim s_j$. $s_i \not\sim s_j$ means that the two posts provide sufficiently different information in order to keep the summaries from being redundant.*

IV. Selected Approaches for Twitter Summaries

Among the many algorithms we discuss in prior papers [1]–[4] for single-length summary creation for tweets, an algorithm called the Hybrid TF-IDF algorithm that we developed worked best. Thus, in this paper, we extend this algorithm to obtain multi-post summaries. The contributions of this paper include introduction of a hybrid TF-IDF based algorithm and a clustering based algorithm for obtaining multi-post summaries of Twitter posts along with detailed analysis of the Twitter post domain for text processing by comparing these algorithms with several other summarization algorithms. We find some unexpected results when we apply multiple document summarization algorithms to short informal documents.

A. Hybrid TF-IDF with Similarity Threshold

Term Frequency Inverse Document Frequency, is a statistical weighting technique that assigns each term within a document a weight that reflects the term's saliency within the document. The weight of a post is the summation of the individual term weights within the post. To determine the weight of a term, we use the formula:

$$TF\_IDF = tf_{ij} * \log_2 \frac{N}{df_j} \quad (1)$$

where tf_{ij} is the frequency of the term T_j within the document D_i, N is the total number of documents, and df_j is the number of documents within the set that contain the term T_j. We assume that a term corresponds to a word and select the most weighted post as summary.

The TF-IDF value is composed of two primary parts. The term frequency component (TF) assigns more weight to words that occur frequently within a document because important words are often repeated. The inverse document frequency component (IDF) compensates for the fact that some words such as common stop words are frequent. Since these words do not help discriminate between one sentence or document over another, these words are penalized proportionally to their inverse document frequency. The logarithm is taken to balance the effect of the IDF component in the formula.

Equation (1) defines the weight of a term in the context of a document. However, a microblog post is not a traditional document. Therefore, one question we must first answer is how we define a document. One option is to define a single document that encompasses all the posts. In this case, the TF component's definition is straightforward since we can compute the frequencies of the terms across all the posts. However, doing so causes us to lose the IDF component since we only have a single document. On the other extreme, we could define each post as a document making the IDF component's definition clear. But, the TF component now has a problem: because each post contains only a handful of words, most term frequencies will be a small constant for a given post.

To handle this situation, we redefine TF-IDF in terms of a hybrid document. We primarily define a document as a single post. However, when computing the term frequencies, we assume the document is the entire collection of posts. Therefore, the TF component of the TF-IDF formula uses the entire collection of posts while the IDF component treats each post as a separate document. This way, we have differentiated term frequencies but also do not lose the IDF component.

We next choose a normalization method since otherwise the TF-IDF algorithm will always bias towards longer posts. We normalize the weight of a post by dividing it by a normalization factor. Since common stop words do not help discriminate the saliency of sentences, we give stop words—as defined by a prebuilt list—a weight of zero. Given this, our definition of the TF-IDF summarization algorithm is now complete for microblogs. We summarize this algorithm below in Equations (2)-(6).

$$W(s) = \frac{\sum_{i=0}^{\#WordsInPost} W(w_i)}{nf(s)} \quad (2)$$

$$W(w_i) = tf(w_i) * \log_2(\text{idf}(w_i)) \quad (3)$$

$$tf(w_i) = \frac{\#OccurrencesOfWordInAllPosts}{\#WordsInAllPosts} \quad (4)$$

$$idf(w_i) = \frac{\#Posts}{\#PostsInWhichWordOccurs} \quad (5)$$

$$nf(s) = \max[MinimumThreshold, \#WordsInPost] \quad (6)$$

where W is the weight assigned to a post or a word, nf is a normalization factor, w_i is the ith word, and s is a post.

We select the top k most weighted posts. In order to avoid redundancy, the algorithm selects the next top post and checks to make sure that it does not have a similarity above a given threshold t with any of the other previously selected posts because the top most weighted posts may be very similar or discuss the same subtopic. This similarity threshold filters out a possible summary post s'_i if it satisfies the following condition:

$$\text{sim}(s'_i, s_j) > t$$

$\forall s_j \in R$ where R is the set of posts aleady chosen for the final summary and t is the similarity threshold. We use the cosine similarity measure. The threshold was varied from 0 to 0.99 in increments of 0.01 for a total of 100 tests in order to find the best threshold to be used.

B. Cluster Summarizer

We develop another method for summarizing a set of Twitter posts. Similar to [15] and [16], we first cluster the tweets into k clusters based on a similarity measure and then summarize each cluster by picking the most weighted post as determined by the Hybrid TF-IDF weighting described in Section IV-A.

During preliminary tests, we evaluated how well different clustering algorithms would work on Twitter posts using the weights computed by the Hybrid TF-IDF algorithm and the cosine similarity measure. We implemented two variations of the k-means algorithm: bisecting k-means [18] and k-means++ [19]. The bisecting k-means algorithm initially divides the input into two clusters and then divides the largest cluster into two smaller clusters. This splitting is repeated until the kth cluster is formed. The k-means++ algorithm is similar to the regular k-means algorithm except that it chooses the initial centroids differently. It picks an initial centroid c_1 from the set of vertices V randomly. It then chooses the next centroid c_i, selecting $c_i = v' \in V$ with the probability $\frac{D(v')^2}{\sum_{v \in V} D(v)^2}$ where $D(v)$ is the shortest Euclidean distance from v to the closest center which is already known. It repeats this selection process until k initial centroids have been chosen. After trying these methods, we found that the bisecting k-means++ algorithm—a combination of the two algorithms—performed the best, even though the performance gain above standard k-means was not very high according to our evaluation methods.

Thus, the cluster summarizer attempts to creat k subtopics by clustering the posts. It then feeds each subtopic cluster to the Hybrid TF-IDF algorithm discussed in IV-A that selects the most weighted post for each subtopic.

C. Additional Summarization Algorithms to Compare Results

We compare the results of summarization of the two newly introduced algorithms with baseline algorithms and well-known multi-document summarization algorithms. The baseline algorithms include a Random summarizer and a Most Recent summarizer. The other algorithms we compare our results with are SumBasic, MEAD, LexRank and TextRank.

1) Random Summarizer: This summarizer randomly chooses k posts or each topic as summary. This method was chosen in order to provide worst case performance and set the lower bound of performance.

2) Most Recent Summarizer: This summarizer chooses the most recent k posts from the selection pool as a summary. It is analogous to choosing the first part of a news article as summary. It was implemented because often intelligent summarizers cannot perform better than simple summarizers that just use the first part of the document as summary.

3) SumBasic: SumBasic [5] uses simple word probabilities with an update function to compute the best k posts. It was chosen because it depends solely on the frequency of words in the original text and is conceptually very simple.

4) MEAD: This summarizer[5] [17] is a well-known flexible and extensible multi-document summarization system and was chosen to provide a comparison between the more structured document domain—in which MEAD works fairly well—and the domain of Twitter posts being studied. In addition, the default MEAD program is a cluster based summarizer so it will provide some comparison to our cluster summarizer.

5) LexRank: This summarizer [7] uses a graph based method that computes pairwise similarity between two sentences—in our case two posts—and makes the similarity score the weight of the edge between the two sentences. The final score of a sentence is computed based on the weights of the edges that are connected to it. This summarizer was chosen to provide a baseline for graph based summarization instead of direct frequency summarization. Though it does depend on frequency, this system uses the relationships among sentences to add more information and is therefore a more complex algorithm than the frequency based ones.

6) TextRank: This summarizer [8] is another graph based method that uses the PageRank [9] algorithm. This provided another graph based summarizer that incorporates potentially more information than LexRank since it recursively changes the weights of posts. Therefore, the final score of each post is not only dependent on how it is related to immediately connected posts but also how those posts are related to other posts. TextRank incorporates the whole complexity of the graph rather than just pairwise similarities.

[5]http://www.summarization.com/mead/

V. Experimental Setup

A. Data Collection

For five consecutive days, we collected the top ten currently trending topics from Twitter's home page at roughly the same time every evening. For each topic, we downloaded the maximum number (approximately 1500) of posts. Therefore, we had 50 trending topics with a set of 1500 posts for each.

B. Preprocessing the Posts

Pre-processing steps included converting any Unicode characters into their ASCII equivalents, filtering out any embedded URL's, discarding spam using a Naïve Bayes classifier, etc. These pre-processing steps and their rationale are described more fully in [1].

C. Evaluation Methods

Summary evaluation is performed using one of two methods: intrinsic, or extrinsic. In intrinsic evaluation, the quality of the summary is judged based on direct analysis using predefined metrics such as grammaticality, fluency, or content [20]. Extrinsic evaluations measure how well a summary enables a user to perform a task. To perform intrinsic evaluation, a common approach is to create one or more manual summaries and to compare the automated summaries against the manual summaries. One popular automatic evaluation metric is ROUGE, which is a suite of metrics [21]. Both precision and recall of the automated summaries can be computed using related formulations of the metric. Given that MS is the set of manual summaries and u is the set of unigrams in a particular manual summary, precision can be defined as

$$p = \frac{\sum_{m \in MS} \sum_{u \in m} \text{match}(u)}{\sum_{m \in MS} \sum_{u \in m} \text{count}(u)} \left(= \frac{matched}{retrieved} \right), \quad (7)$$

where $count(u)$ is the number of unigrams in the automated summary and $match(u)$ is the number of co-occurring unigrams between the manual and automated summaries. The ROUGE metric can be slightly altered so that it measures the recall of the auto summaries such that

$$r = \frac{\sum_{m \in MS} \sum_{u \in m} \text{match}(u)}{\mid MS \mid * \sum_{u \in a} \text{count}(u)} \left(= \frac{matched}{relevant} \right), \quad (8)$$

where $\mid MS \mid$ is the number of manual summaries and a is the auto summary. We also report the F-measure, which is the harmonic mean of precision and recall.

Lin's use of of ROUGE with the very short (around 10 words) summary task of DUC 2003 shows that ROUGE-1 and other ROUGEs correlate highly with human judgments [21]. Since this task is very similar to creating microblog summaries, we implement ROUGE-1 as a metric. However, since we want certainty that ROUGE-1 correlates with a human evaluation, we implemented a human evaluation using Amazon Mechanical Turk[6], a paid system that pays human workers small amounts of money for completing a short Human Intelligence Task, or HIT. The HITs used for summary evaluation displayed the summaries to be compared side by side with the topic specified. Then, we asked the user, "The auto-generated summary expresses _______ of the meaning of the human produced summary." The possible answers were "All," "Most," "Some," "Hardly Any" and "None" which correspond to a score of 5 through 1, respectively.

[6] http://www.mturk.com

TABLE I
Answers to the survey about how many clusters seemed appropriate for each Twitter topic.

Answer	"3 (Less)"	"4 (About Right)"	"5 (More)"
Count	13	28	9

D. Manual Summarization

1) Choice of k: An initial question that we must answer before using any multi-post extractive summarizer on a set of Twitter posts is the question of how many posts are appropriate in a summary. Though it is possible to choose k automatically for clustering [22], we decided to focus our experiments on summaries with a predefined value of k for several reasons. First, we wanted to explore other summarization algorithms for which automatically choosing k is not as straightforward as in the cluster summarization algorithm. For example, the SumBasic summarization does not have any mechanism for choosing the right number of posts in the summary. Second, we thought it would be difficult to perform evaluation where the manual summaries were two or three posts in length and the automatic summaries were five or six posts in length—or vice versa—because the ROUGE evaluation metric is sensitive to length even with some normalization.

To get a subjective idea of what people thought about the value of $k = 4$ after being immersed in manual clustering for a while, we took a survey of the volunteers after they performed clustering of 50 topics—2 people for each of the 25 topics—with 100 posts in each topic. We asked them "How many clusters do you think this should have had?" with the choices "3 (Less)", "4 (About Right)" or "5 (More)". The results are in Table I. This survey is probably biased towards "4 (About Right)" because the question does not allow for numbers other than 3, 4 or 5. Therefore, these results must be taken tentatively but they at least suggest that there is some significant variability about the best value for k. Our bias is also based on the fact that our initial 1500 Twitter posts on each topic were obtained within a small interval of 15 minutes so we thought a small number would be good.

Since the volunteers had already clustered the posts into four clusters, the manual summaries were four-post long as well. This kept the already onerous manual summary creation process somewhat simple. However, this also means that being dependent on a single length for the summaries may impact our evaluation process described next in an unknown way.

2) Manual Summarization Method: Our manual multi-post summaries were created by volunteers who were undergraduates from around the US gathered together in an NSF-supported REU program. Each of the first 25 topics was manually summarized by two different volunteers[7] by performing

[7] A total of 16 volunteers produced manual summaries in such a combination that no volunteer would be compared against another specified volunteer more than once.

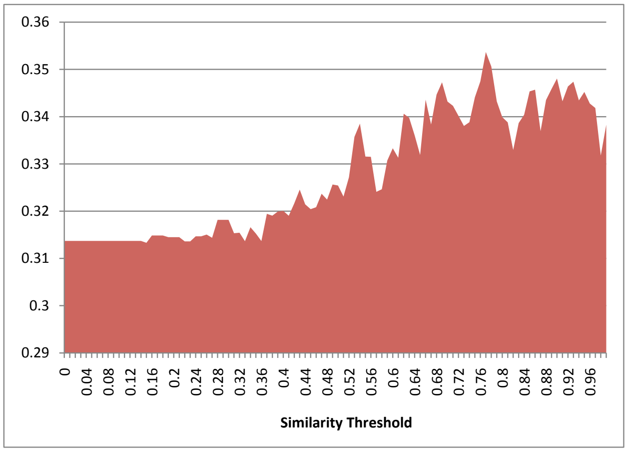

Fig. 1. F-measures of Hybrid TF-IDF algorithm over different thresholds.

steps parallel to the steps of the cluster summarizer. First, the volunteers clustered the posts into 4 clusters ($k = 4$). Second, they chose the most representative post from each cluster. And finally, they ordered the representative posts in a way that they thought was most logical or coherent. These steps were chosen because it was initially thought that a clustering based solution would be the best way to summarize the Twitter posts and it seemed simpler for the volunteers to cluster first rather than simply looking at all the posts at once. These procedures probably biased the manual summaries—and consequently the results—towards clustering based solutions but since the cluster summarizer itself did not perform particularly well in the evaluations, it seems that this bias was not particularly strong.

E. Setup of the Summarizers

Like the manual summaries, the automated summaries were restricted to producing four post summaries. For MEAD, each post was formatted to be one document. For LexRank—which is implemented in the standard MEAD distribution—the posts for each topic were concatenated into one document. Because the exact implementation of TextRank [8] was unavailable, the TextRank summarizer was implemented internally.

For the Hybrid TF-IDF summarizer, in order to keep the posts from being too similar in content, a preliminary test to determine the best cosine similarity threshold was conducted. The F-measure scores when varying the similarity threshold t of the Hybrid TF-IDF summarizer from 0 to 0.99 are shown in Figure 1. The best performing threshold of $t = 0.77$ seems to be reasonable because it allows for some similarity between final summary posts but does not allow them to be nearly identical.

VI. Results and Analysis

The average F-measure of all the iterations was computed. For the summarizers that involve random seeding (e.g., random summarizer and cluster summarizer), 100 summaries were produced for each topic to avoid the effects of random seeding. These numbers can be seen more clearly in Table II. Also, because we realized that the overlap of the topic keywords in the summary is trivial since every post contains the keywords, we ignored keyword overlap in our ROUGE calculations.

For the human evaluations using Amazon Mechanical Turk, each automatic summary was compared to both manual summaries by two different evaluators. This leads to 100 evaluations per summarizer as can be seen in Table II. The manual summaries were evaluated against each other by pretending that one of them was the automatic summary.

TABLE II
Evaluation Numbers for ROUGE and MTurk Evaluations.

Number of summaries	Randomly seeded*	Others
Number of topics	25	25
Summaries per topic	100	1
Total summaries computed	**2500**	**25**
ROUGE evaluation		
ROUGE scores computed	**2500**	**25**
MTurk evaluation		
Number of summaries evaluated	25+	25
Number of manual summaries per topic	2	2
Evaluators per manual summary	2	2
Total MTurk evaluations	**100**	**100**

* The randomly seeded summaries were the Random Summarizer and the Cluster Summarizer.
+An average scoring post based on the F-measure for each topic was chosen for the MTurk evaluations because evaluating 2500 summaries would have been impractical.

TABLE III
Average values of F-measure, recall and precision ordered by F-measure.

	F-measure	Recall	Precision
LexRank	0.2027	0.1894	0.2333
Random	0.2071	0.2283	0.1967
Mead	0.2204	0.3050	0.1771
Manual	0.2252	0.2320	0.2320
Cluster	0.2310	0.2554	0.2180
TextRank	0.2328	0.3053	0.1954
MostRecent	0.2329	0.2463	0.2253
Hybrid TF-IDF	0.2524	0.2666	0.2499
SumBasic	0.2544	0.3274	0.2127

A. Results

Our experiments evaluated eight different summarizers: random, most recent, MEAD, TextRank, LexRank, cluster, Hybrid TF-IDF and SumBasic. Both the automatic ROUGE based evaluation and the MTurk human evaluation are reported for all eight summarizers in Figures 2 and 3, respectively. The values of average F-measure, recall and precision can be seen in Table III. The values of average MTurk scores can be seen at the top of Table V.

B. Analysis of Results

1) General Observations: We see that both the ROUGE scores and the human evaluation scores do not seem to obviously differentiate among the summarizers as seen in Figures 2 and 3. Therefore, we performed a paired two-sided T-test for each summarizer compared to each other summarizer for both the ROUGE scores and the human evaluation scores. For the ROUGE scores, the twenty five average F-measure scores corresponding to each topic were used for the paired

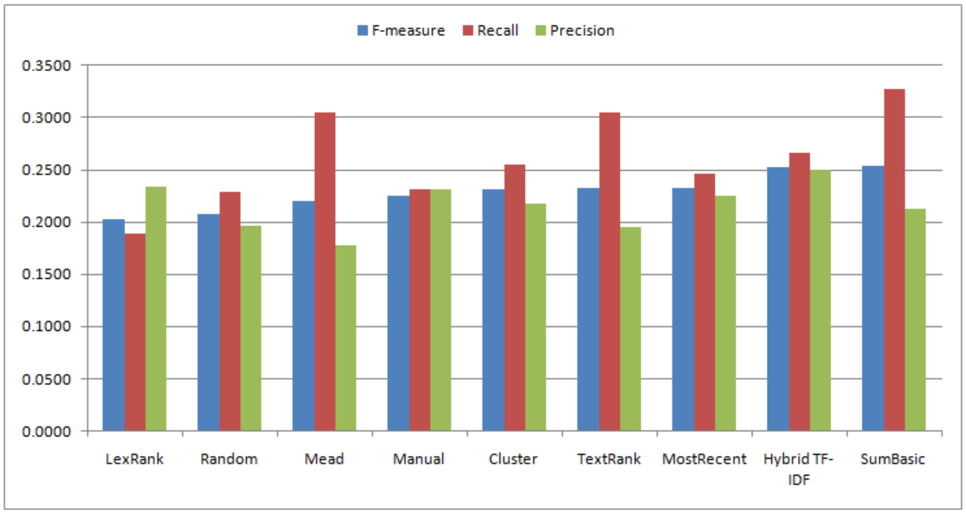

Fig. 2. Average F-measure, precision and recall ordered by F-measure.

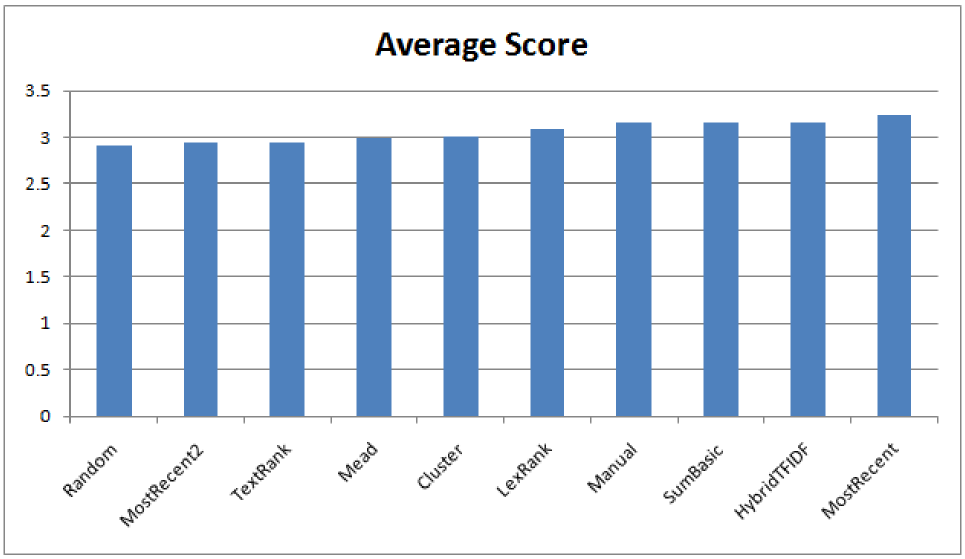

Fig. 3. Average scores for human evaluation using Amazon Mechanical Turk ordered by average score.

T-test and for the human evaluation, all hundred evaluation scores ordered by topic and then by score were used for the paired T-test. The pairwise matrix of p-values for these tests as well as the average score for each summarizer can be seen in Tables IV and V. The bolded p-values indicate that the summarizers are statistically different at the 95% confidence level.

Since the number of unigrams in the automated summary could significantly affect the ROUGE scores, the average number of characters for each summarizer is shown in Figure 4. The high average number of characters for the MEAD, TextRank and SumBasic summarizers—approximately 50% higher than the manual summaries—explains why the recall values of the MEAD, TextRank and SumBasic summarizers are particularly high. In addition, the results help explain why the recall of every summarizer except the LexRank summarizer isfis not pa higher than their corresponding precision measures since the average number of characters for all the other summarizers is greater than the average number of characters for the manual summaries.

Overall, it seems from these results that the simple frequency based summarizers, namely SumBasic and Hybrid TF-IDF, perform better than summarizers that incorporated more information or more complexity such as LexRank, TextRank or MEAD. This probably has much to do with the special nature of Twitter posts in which posts often have very little structure and have so few words that forming relationships between pairs of posts is not particularly helpful. In addition, since each post is mostly uncorrelated with any other post—except for replies and retweets—, thematic or progressive development of a topic is rare, and therefore, more complex relational models will probably not capture more topical information than frequency methods. More specific analysis on each of the summarizers is described in the following sections.

a) Manual Summaries: Though the manual to manual F-measure scores seem low at 0.3291, this may be explained by several factors. First, the instructions given to the volunteers for summarizing did not give any guidelines on how to cluster

TABLE IV
P-VALUES FOR TWO-SIDED PAIRED T-TEST FOR F-MEASURES IN THE EXPERIMENTS.

Avg. F-meas.	0.187	0.207	0.220	0.225	0.232	0.233	0.233	0.252	0.254
	LR	Rand.	Mead	Man.	Clust.	TR	MR	Hyb.	SB
LexRank		0.102	**0.040**	0.140	**0.004**	**0.010**	**0.043**	**0.003**	**0.001**
Random	0.102		0.226	0.318	**0.014**	**0.021**	0.162	**0.011**	**0.002**
Mead	**0.040**	0.226		0.789	0.364	0.408	0.554	0.109	**0.027**
Manual	0.140	0.318	0.789		0.714	0.644	0.783	0.229	0.141
Cluster	**0.004**	**0.014**	0.364	0.714		0.946	0.963	0.095	0.069
TextRank	**0.010**	**0.021**	0.408	0.644	0.946		0.995	0.329	0.098
MostRecent	**0.043**	0.162	0.554	0.783	0.963	0.995		0.429	0.333
HybridTFIDF	**0.003**	**0.011**	0.109	0.229	0.095	0.329	0.429		0.920
SumBasic	**0.001**	**0.002**	**0.027**	0.141	0.069	0.098	0.333	0.920	

TABLE V
P-VALUES FOR TWO-SIDED PAIRED T-TEST FOR HUMAN EVALUATION FOR THE EXPERIMENTS.

Avg. Score	2.91	2.94	2.94	3.00	3.01	3.09	3.15	3.15	3.16	3.24
	Rand.	MR2	TR	Mead	Clus.	LR	Man.	SB	Hyb.	MR
Random		0.801	0.765	0.331	0.305	**0.049**	**0.019**	**0.011**	**0.007**	**0.000**
MostRecent2*	0.801		1.000	0.615	0.531	0.156	0.058	0.077	0.072	**0.011**
TextRank	0.765	1.000		0.534	0.461	0.116	**0.042**	**0.048**	**0.030**	**0.005**
Mead	0.331	0.615	0.534		0.919	0.314	0.108	0.096	0.103	**0.012**
Cluster	0.305	0.531	0.461	0.919		0.407	0.167	0.123	0.092	**0.021**
LexRank	**0.049**	0.156	0.116	0.314	0.407		0.463	0.488	0.461	0.104
Manual	**0.019**	0.058	**0.042**	0.108	0.167	0.463		1.000	0.921	0.358
SumBasic	**0.011**	0.077	**0.048**	0.096	0.123	0.488	1.000		0.910	0.307
HybridTFIDF	**0.007**	0.072	**0.030**	0.103	0.092	0.461	0.921	0.910		0.304
MostRecent*	**0.000**	**0.011**	**0.005**	**0.012**	**0.021**	0.104	0.358	0.307	0.304	

* Please see Section VI-B1c for an explanation of the two different results for the Most Recent summarizer.

the posts except whatever themes or subtopics the volunteers thought could be good clusters. Therefore, the clusters for a topic may have been significantly different from one person to another depending on how they wanted to differentiate the posts. Second, some topics only had thematic overlap rather than unigram overlap. For example, the topic "#MM" was a topic that stood for "Music Mondays" and the tweets would simply have names of songs or names of artists. Obviously, the names of songs or artists do not tend to overlap naturally. In addition, these results seem to agree with the fairly low F-measure scores computed for one sentence summaries in [1]–[4].

It may seem odd that the manual to manual scores are actually lower than some of the other summarizers' scores. However, this is possible with the F-measure scores because if the summarizer produced a summary that was very similar to one of the manual summaries, it would score a very high F-measure compared to the first manual summary. Then, when compared to the second summary, it could also be decently similar since it is not exactly like the other manual summary. In this way, F-measure scores could be higher than the manual to manual F-measure scores. In a similar manner, the human evaluation scores that are higher on average than manual could mean that the generated summaries captured some of the good ideas from both manual summaries better than each manual summary captured the ideas of the other manual summary.

b) Random and Most Recent Summarizer: The seemingly high F-measure and human score of the random summarizer may be explained by a few characteristics of microblog posts. First, microblog posts on a given topic tend to use similar words so a fair amount of incidental word and theme overlap seems reasonable. Second, a particularly interesting post on a given topic can often be quoted verbatim by many other microblog users so the random summarizer has a better chance of selecting one of these informative retweeted posts. Third, the preprocessing of the posts helped reduce the original set of posts to a set of less noisy posts.

c) Most Recent Summarizer: Because the test posts were collected by getting posts for trending topics—very active topics—, most of the posts for each topic were generated within fifteen minutes of each other or even less. Therefore, it seemed unlikely that the most recent posts would be particularly more relevant than random posts. The p-values for the F-measure scores seem to agree with only a p-value of 0.162 when comparing the Random summarizer to the most recent summarizer. However, the human scores showed that maybe the most recent summarizer was better than random. One possible reason for the most recent summarizer doing better than expected is that the manual summaries may be biased towards the more recent posts because these were displayed to the volunteers first and may have biased their judgments of the best posts to use for the summary.

Initially, the first set of human scores for evaluating the most recent summarizer—the results marked "MostRecent"—was surprisingly higher than expected outperforming SumBasic and Hybrid TF-IDF even though its F-measure scores were significantly lower than SumBasic or Hybrid TF-IDF. Because of this, we considered that maybe the results were skewed by one or two Mechanical Turk workers who rated a few of the posts significantly higher than most people would.

Therefore, we decided to retrieve a second set of one hundred evaluations with essentially the same procedure—the

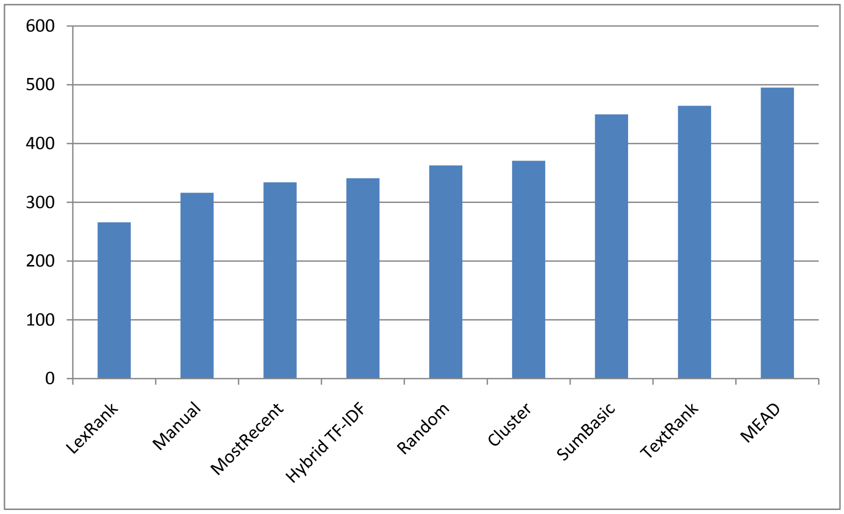

Fig. 4. Average number of characters for each summarizer.

results marked "MostRecent2"—and got significantly lower scores than the previous time. In fact, the p-value for comparing the two sets of results is 0.011. Therefore, it seems that we may have been correct that the first set of results was skewed. Even if both results are valid, the average of both result sets would be 3.09, which is a little lower than the manual scores. Therefore, in general, it seems that the most recent summarizer does not perform better than the random summarizer. This would agree with our original idea that the most recent posts should not be inherently more relevant than random posts because of the nature of Twitter posts being generated generally haphazardly without order or specific thematic development.

d) Frequency Based Summarizers (SumBasic and Hybrid TF-IDF): The simple frequency based summarizers seemed to outperform all other algorithms both in F-measure scores and human evaluation scores. For the F-measure scores, both are significantly different from the LexRank and Random summarizers, and SumBasic is significantly different than the MEAD summarizer. For the human evaluation scores, both are significantly different from Random and TextRank.

The Hybrid TF-IDF summarizer can be seen as adding a little complexity to the simple SumBasic algorithm by including information regarding the IDF component of the term frequency calculation. From the results, it seems that this added complexity is not particularly helpful in computing summaries. However, it should be noted that the Hybrid TF-IDF has a closer balance between precision and recall whereas the SumBasic algorithm has a higher recall than precision. This suggests that the SumBasic algorithm may be biased towards longer summaries but does not necessarily affect its performance or overall usefulness.

Both algorithms employ a redundancy reduction method in order to avoid summaries that have very similar posts in them. In addition, both use a fairly greedy approach to this reduction by selecting the next best post—either the best weighted non-redundant post in the case of Hybrid TF-IDF or the best weighted post after weight recalculation in SumBasic. Therefore, it seems that simple word frequency calculations and redundancy reduction are particularly important for summarizing Twitter topics.

e) Cluster Based Summarizers (MEAD and Cluster): The two cluster based summarizers—the baseline MEAD and our implementation of a cluster summarizer—did not do as well as expected. They performed significantly better than the Random summarizer in the F-measure scores but did not perform significantly better in the human evaluation. Like with the frequency summarizers, these summarizers attempted to reduce redundancy but did so by clustering the posts first and then summarizing based on these clusters. However, clustering did not seem to increase performance. This could be true because of the short, unstructured and informal nature of Twitter posts that do not correlate with the expectations of more traditional techniques for summarizing that use clustering. Also, the posts may be particularly difficult to cluster unlike more structured or longer document collections because they have so few non-zero features. Since the default MEAD summarizer has much better recall than precision, it may be improved upon by normalizing the weights of sentences more strongly than the default.

f) Graph Based Summarizers (LexRank and TextRank): The results of the two graph based summarizers—LexRank and TextRank—were intriguing. In the F-measure scores, LexRank did worse than random but TextRank did decently well by at least being significantly different than random. However, in the human scores, TextRank did not even perform significantly better than random whereas LexRank performed significantly better than random. One interesting aspect that may explain some of this is that LexRank was the only algorithm that had a better precision than recall. This may suggest that LexRank in general chose shorter more concise posts for its summaries. TextRank was the opposite by having a much higher recall than precision. Because the human evaluation scores suggest that LexRank performed better than TextRank, these algorithms may suggest that the F-measure score is biased towards longer summaries.

In general, however, since the frequency based summarizers did better—and in a several cases significantly better—than the graph based summarizers, it seems that the added complexity of interrelationships did not help in summarizing Twitter posts.

VII. Conclusion

Our conclusions, based on results reported in this paper, is that the simple frequency based summarizers—Hybrid TF-IDF and SumBasic—produced the best results both in F-measure scores and in human evaluation scores. Because the more complex algorithms did not perform as well, it seems that simple word frequency and redundancy reduction are the best techniques for summarizing Twitter topics. This is most likely due to the unstructured, unconnected and short characteristics of Twitter posts that are not like traditional documents.

This project could be further extended in many ways. Other ways of evaluating microblog summaries could be considered that would focus on extrinsic evaluations such as asking users to specify the helpfulness of a summary. Methods for dynamically discovering a good value for k for k-means or other agglomerative clustering algorithms for a given topic for multi-post summarization could be researched. Researchers have demonstrated that for (multi-document) summarization, it is beneficial to being able to detect named entities and events in the documents as well as to be able to parse, at least identify noun phrases and their heads. The tweets may have to be standardized to their normal English syntax before parsers will work (e.g., see [23]). That is why, for the current paper, we had decided to use simple word similarity-based non-hierarchical clustering algorithms. In addition, based on our experience that simple algorithms seem to perform well with Twitter summarization, it is not clear that added complexity will improve the quality of the summaries.

Acknowledgment

The work reported in this paper has been partially supported by NSF grants ARRA: CNS-0851783 and CNS-0958576.

References

[1] B. Sharifi, "Automatic microblog classification and summarization," Master's thesis, University of Colorado at Colorado Springs, 2010.

[2] B. Sharifi, M.-A. Hutton, and J. K. Kalita, "Summarizing microblogs automatically," in *NAACL-HLT*, 2010, pp. 685–688.

[3] ——, "Experiments in microblog summarization," in *IEEE SocialCom 2010*, 2010, pp. 49–56.

[4] ——, "Automatic summarization of twitter topics," in *National Workshop on Design and Analysis of Algorithms, Tezpur, India*, 2010, pp. 121–128.

[5] L. Vanderwende, H. Suzuki, C. Brockett, and A. Nenkova, "Beyond SumBasic: Task-focused summarization with sentence simplification and lexical expansion," *Information Processing & Management*, vol. 43, no. 6, pp. 1606–1618, 2007.

[6] D. Radev, S. Blair-Goldensohn, and Z. Zhang, "Experiments in single and multi-document summarization using mead," *DUC-01*, vol. 1001, p. 48109, 2001.

[7] G. Erkan and D. Radev, "Lexrank: graph-based centrality as salience in text summarization," *Journal of Artificial Intelligence Research*, vol. 22, pp. 457–480, 2004.

[8] R. Mihalcea and P. Tarau, "TextRank: Bringing order into texts," in *EMNLP*. Barcelona: ACL, 2004, pp. 404–411.

[9] S. Brin and L. Page, "The anatomy of a large-scale hypertextual Web search engine* 1," *Computer networks and ISDN systems*, vol. 30, no. 1-7, pp. 107–117, 1998.

[10] A. Kolcz, V. Prabakarmurthi, and J. Kalita, "Summarization as feature selection for text categorization," in *Proceedings of the tenth international conference on Information and knowledge management*. ACM, 2001, pp. 365–370.

[11] V. Ganti, J. Gehrke, and R. Ramakrishnan, "Cactus—clustering categorical data using summaries," in *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY, USA: ACM, 1999, pp. 73–83.

[12] J. Kalita, *Generating Summary Responses to Natural Language Database Queries*. University of Saskatchewan, Dept. of Computational Science, 1984.

[13] J. Kalita, M. Colbourn, and G. McCalla, "A response to the need for summary responses," in *Proceedings of the 10th International Conference on Computational Linguistics and 22nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 1984, pp. 432–436.

[14] J. Kalita, M. Jones, and G. McCalla, "Summarizing natural language database responses," *Computational Linguistics*, vol. 12, no. 2, pp. 107–124, 1986.

[15] K. McKeown, J. Klavans, V. Hatzivassiloglou, R. Barzilay, and E. Eskin, "Towards multidocument summarization by reformulation: Progress and prospects," in *AAAI*, 1999, pp. 453–460.

[16] I. Dhillon, "Co-clustering documents and words using bipartite spectral graph partitioning," in *ACM SIGKDD*. ACM, 2001, pp. 269–274.

[17] D. Radev, T. Allison, S. Blair-Goldensohn, J. Blitzer, A. Çelebi, S. Dimitrov, E. Drabek, A. Hakim, W. Lam, D. Liu, J. Otterbacher, H. Qi, H. Saggion, S. Teufel, M. Topper, A. Winkel, and Z. Zhang, "Mead - a platform for multidocument multilingual text summarization," in *LREC 2004*, Lisbon, Portugal, May 2004.

[18] Y. Zhao and G. Karypis, "Criterion functions for document clustering: Experiments and analysis," 2001. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.17.3151

[19] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *ACM-SIAM symposium on Discrete algorithms*, Philadelphia, PA, USA, 2007, pp. 1027–1035.

[20] J. Lin, M. Ozsu, and L. Liu, "Summarization," *Encyclopedia of Database Systems, Springer*, 2009.

[21] C.-Y. Lin and E. Hovy, "Automatic evaluation of summaries using n-gram co-occurrence statistics," in *NAACL '03*. Morristown, NJ, USA: Association for Computational Linguistics, 2003, pp. 71–78.

[22] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a data set via the gap statistic," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, no. 2, pp. 411–423, 2001.

[23] J. Kaufmann and J. Kalita, "Syntactic normalization of twitter messages," in *International Conference on Natural Language Processing (ICON 2011)*. New Delhi: McMillan, India, 2010, pp. 149–158.