Automatic Summarization of Twitter Topics

Beaux Sharifi, Mark-Anthony Hutton and Jugal Kalita
University of Colorado at Colorado Springs
1420 Austin Bluffs Parkway
Colorado Springs, CO 80918
{bsharifi, mhutton, jkalita}@uccs.edu

Abstract

During recent years, socially generated content has become pervasive on the World Wide Web. The enormous amount of content generated in blog sites, social networking sites such as Facebook and Myspace, encyclopedic sites such as Wikipedia, has not only empowered ordinary users of the Web but also contributed to the vastness as well as richness of the Web's contents. In this paper, we focus on a recent trend called microblogging, and in particular a site called Twitter that allows a huge number of users to contribute frequent short messages. The content of such a site is an extra-ordinarily large number of small textual messages, posted by millions of users, at random or in response to perceived events or situations. However, out of such random and massive disorganization of messages usually trends emerge as a large number of users post similar messages on similar topics. These trends can be discovered using statistical analysis of mass of posts. We have developed an algorithm that takes a trending phrase or any phrase specified by a user, collects a large number of posts containing the phrase, and provides an automatically created summary of the posts related to the term. We present examples of summaries we produce along with initial qualitative evaluation. It is possible to get a global view of the content of the text message repository in terms of a set of short summaries of trending terms during the course of a period of time such as an hour or a day.

Introduction

Since Twitter's inception in 2006, it has grown at an unprecedented rate. In just over three years, the microblogging service has grown to almost 6 million unique visitors each month (Kazeniac 2009). According to Pingdom, users are "tweeting", or sending short 140-character messages to Twitter, approximately 27 million times a day (Schonfeld 2009). While the majority of these tweets are pointless babble or conversational, approximately 3.6% of these posts are topics of mainstream news, and another 8.7% are topics interesting enough for users to forward to their own followers via re-tweeting (Schonfeld 2009). With approximately a million news posts being sent a day as well as many other sources of information, Twitter has become an important source of gathering real-time information on almost any topic imaginable. For example, Twitter has been cited as breaking many important events before traditional media such as the attacks in Mumbai (Beaumont 2008) and the crash of the US Airways flight into the Hudson River (Beaumont 2009). With important news being sent in real time to Twitter, it is important that users of Twitter are able to find these important tweets as they occur without having to sort through all the other irrelevant information being posted.

In order to help users sort through the vast number of tweets that occur each day, Twitter.com has added a number of tools that help users find the most important topics and their associated tweets. Twitter's homepage displays these important topics – which are just short phrases such as "New Moon", "Glee", or "Janet Jackson" – for three different ranges of time in order to see what topics are popular at the moment, during the present day, or over the last week. For most topics, users are forced to read through related posts (by clicking on the topic) in order to try and understand why a topic is trending. This process is tedious and error prone as returned posts are prioritized only by recency. Therefore, for a given topic, users are likely to encounter spam, posts in other languages, rants, and other sources of misinformation. To help users further, Twitter has partnered with the third-party website WhatTheTrend[1] in order to provide definitions or trending topics. WhatTheTrend allows users to manually enter descriptions of why a topic is trending. While the idea is good in theory, in practice WhatTheTrend also suffers with spam and rants as users are free to enter whatever definition they prefer for a trending topic. A quick look at the history of definitions for a few topics on WhatTheTrend often shows definitions oscillating between users trying to spam the website and other users trying to provide accurate information. The biggest drawback to the site is definitions are entered by hand and not automated. Therefore, there is often some lag time before a new trending topic is defined by a user which defeats the purpose of Twitter trying to be a source of real time news.

While WhatTheTrend is a step in the right direction, what is really needed is an automated technique that can summarize important events as they occur in real time. We have developed such a method that can summarize trending topics from Twitter into short one-line summaries in real time. By using Twitter's own API, we are able to retrieve trending topics along with their related posts, and summarize these posts into a convenient summary. Our results show that our automated summarizer produces summaries that are very close to human-generated summaries for the same set of posts. For example, the following are a sample of automatically produced summaries for some recently trending topics on Twitter:

Topic	Automated Summary	Date Acquired
Captain Lou	Wrestler, personality Captain Lou Albano dies at 76	10/14/2009
Dow Jones	The Dow Jones Industrial Average passes 10,000 for the first time since October 7th, 2008.	10/14/2009
Dodgers	Phillies defeat Dodgers to take the National League Championship series.	10/21/2009
Pirate	Me hearties, today is the 14th annual international Talk Like a Pirate Day!	09/19/2009
Limbaugh	Limbaugh dropped from group bidding for St. Louis Rams	10/14/2009
G20	Trouble breaks out at G20 summit: Protesters and riot police have clashed ahead of the G20 summit in Pittsburgh	09/24/2009
AT&T	AT&T plans for iPhone MMS to arrive Friday	09/23/2009
Bloomberg	Bloomberg Acquires Businessweek for Less Than $5 million	10/13/2009

Table 1. Table demonstrating automatically generated summaries for several recently trending topics on Twitter.

[1] http://www.whatthetrend.com

Related Work

Automatic summarization of documents has been a topic of research for a long time. Some very early work focused on summarizing results of database queries to make them more suitable for presentation during natural language interactions (e.g., Kalita 1984, Kalita et al. 1984, Kalita et al. 1986). A lot of later work has focused on summarizing electronic documents such as articles or academic papers. Most summaries are generated for the purposes of providing a "gist" of a document or a set of documents to human readers (e.g., Luhn 1958, Edmundson 1969, Brandow et al. 1995). Mahesh (1997) summarizes Web documents for fast document browsing. Summaries are sometimes also used as inputs to machine learning approaches, say for categorization. For example, Kolcz et al. (2001) look at the problem summarizing textual documents in order to classify them more efficiently, considering the summaries as a feature to be input to a classifier using SVMs. Most such studies used a news corpus like the Reuters dataset. As the Web started growing in size, the focus moved to Web pages. For example, Mahesh (1997) examines the effectiveness of Web document summarization by sentence extraction. Recently, there has been work on summarizing blogs (e.g. Zhou and Hovy 2006, Hu et al. 2007, Lin and Sundaram 2007, Bossard et al. 2008). Blogs can be of any size. Current examples of automatic summarizing of blogs focuses on one of two techniques: summarization via abstraction or summarization via extraction (Kolcz et al. 2001). Abstraction is difficult to automate since it attempts to generate summaries that are paraphrases of a set of documents. For this reason, most techniques focus on extraction: the selecting of salient pieces of documents in order to generate a summary. Applying extraction on microblogs at first appears irrelevant since a microblog post is already shorter than most summaries. However, extraction is possible when one considers extracting from multiple microblogs posts that are all related to a central theme. We develop such an extraction technique that selects phrases from one or more microblog posts through the analysis of common word usage across multiple microblog posts that are all related by a common phrase.

Twitter API

Automatically summarizing trending topics on Twitter requires being able to access Twitter data in real-time. Fortunately, Twitter has promoted third-party applications by providing programming interfaces to its website since almost its inception. Through an entirely HTTP-based API, users can perform almost any task that can be performed via the website's user-interface. For example, one can retrieve trending topics, perform searches, and make modifications to their own blog (Molina 2009). In order to prevent users from monopolizing Twitter's servers, the website has imposed "pagination limits" which restrict the number of requests that can be sent to Twitter. For non-whitelisted users, twitter restricts a user to 150 requests per hour (Molina 2009). Furthermore, searches are limited to returning 1500 posts for a given request. If users wish to have more data, they can apply to be added to Twiter's whitelist which allows receiving a larger stream of the public updates being sent to Twitter in real-time (Kalucki 2009).

While Twitter allows receiving streaming data, we restrict our usage of Twitter's API to receiving trending topic phrases and the related posts around a particular topic. While these API's are limited to a maximum of 1500 posts for a particular topic, our summarizer has been shown to produce comparable automated summaries to human summaries with as few as 100 posts. Furthermore, because of its design, its performance improves as the number of available posts increases.

Phrase Reinforcement Algorithm

As mentioned earlier, the Twitter API returns up to 1500 posts that contain a search phrase. Therefore, given a trending topic (also available via the API), one can query Twitter.com for a set of posts that contain the trending topic phrase. Presently, users would have to read these posts manually in order to comprehend and summarize their content. Instead, we have chosen to automate this process using the following algorithm, which we call the Phrase Reinforcement Algorithm.

The Phase Reinforcement (PR) Algorithms works as follows. The algorithm begins with a starting phrase, which is the topic for which one desires to generate a summary. These are typically a trending topic, but can be other non-trending topics as well. Given the starting phrase, the PR algorithm submits a query to Twitter.com for a list of posts that contain the phrase. If the topic has been discussed on Twitter recently, Twitter will usually be able to return the maximum of 1500 posts that contain the search phrase. Otherwise, Twitter may return less than 1500 posts or even none at all. Given the returned set of posts, the algorithm next filters the posts to remove any spam or irrelevant posts. Filtering is an important step since spam and other irrelevant posts can mislead the PR algorithm into summarizing the spam instead of the desired content. The spam is filtered using a Naïve Bayes classifier which we trained using previously gathered spam content from Twitter.com. We also remove any non-English posts as well as duplicate posts since we are concerned with English summaries only and want to prevent a single user from dominating a topic.

Once we have a set of relevant posts (which we will call the training posts), the PR algorithm formally begins. The central idea of the PR algorithm is to build an ordered acyclic graph of all the words within the set of training posts to the algorithm. The graph is organized around a central root node, which contains the starting phrase we are trying to summarize. Adjacent to the starting node are words that occur either immediately before or after the starting phase within each of the training posts. These adjacent words are also placed either before or after the starting node respective of the ordering found within the training posts. Continuing in this same fashion, adjacent to these adjacent nodes are words that immediately proceed or follow these nodes within the same set of available posts and so on. Because of the ordering of the nodes reflect the ordering of words found within the training posts, every path within the resulting graph starting from the root node will represent an actual phrase found within at least N of the entered posts (N is a parameter to the algorithm). Next, as the graph is constructed, the nodes are weighted according to their frequency of occurrence respective of their word ordering from the root. Therefore, if a word occurs M times after the starting phrase within the training posts, its weight will be proportional to M. After all training posts have been processed and the graph is fully constructed and weighted, the PR algorithm is considered trained and ready to begin generating summaries of the starting phrase.

Once the graph is built, the PR algorithm begins searching for the best partial summary by summing the weight of every unique path starting from the root node to each of the leaf nodes. The path with the most weight is considered the best partial summary path from the root node. We call these partial summaries since the words along these paths only represent one half of our summary since each path either begins or ends with our initial phrase. In order to build the remaining half, we repeat the above algorithm as follows. First, we filter our set of training posts to those that contain our partial summary. Next, we construct a new graph using our filtered training posts and our partial summary phrase as our initial root node. Furthermore, since our partial summary represents one half of our summary, we only build the new

graph in the opposite remaining direction. Once the graph is constructed, all paths within the graph (starting from any node) will represent an actual phrase found within one of our training posts. We then again search each of these paths for the one with the greatest total weight. The words along this path become our final summary.

Figures 1-3 below demonstrate the construction of the two graphs using the following set of training posts:

- "A torch extinguished: Ted Kennedy dead at 77.''
- "A legend gone: Ted Kennedy died of brain cancer."
- "Ted Kennedy was a leader."
- "Ted Kennedy died today."

Each node in the figures below is annotated with the word phrase and its frequency within the set of available posts. The root node has a frequency of zero because it is not counted and node weights are not shown for brevity. For this simple example, assume that a node's weight is equal to its frequency. The green path represents the path with the most total weight and is selected as the resulting summary by the Phrase Reinforcement algorithm. Note that we are requiring a node frequency of at least two before being included as part of the summary. This is an intuitive constraint because we are trying to find a consensus phrase across the available posts in which to build a summary. In practice, we would not add any nodes to the graph whose frequency was less than two. However, we include these nodes in the figures for clarity of the graph structure. Finally, we give some pseudo code of the PR algorithm in Figure 4.

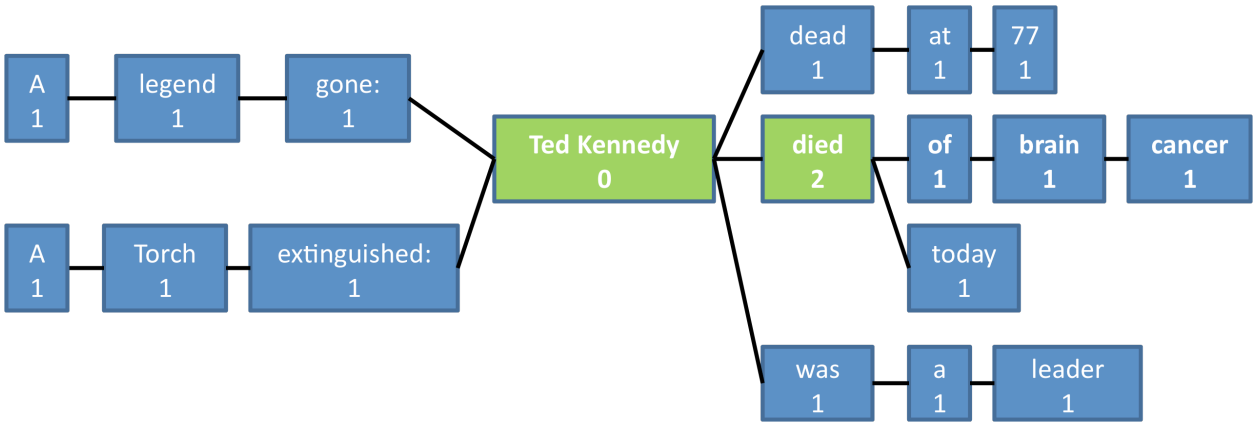

Figure 1. Best partial summary with the initial phrase "Ted Kennedy".

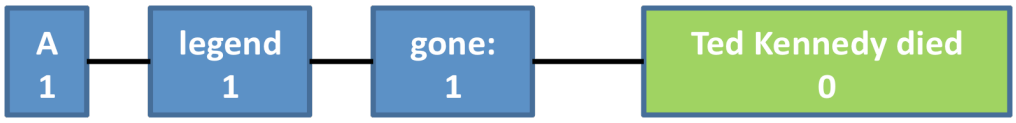

Figure 2. Best partial summary ending with the phrase "Ted Kennedy died".

Figure 3. Best final summary of phrases containing "Ted Kennedy".

i. Gather a set of posts that each contain a common root phrase. These posts should not include spam nor duplicate posts from a single user.
ii. Initialize a graph with a single root node containing the root phrase.
iii. For each post do
    a. For each word *before* the root phrase, beginning with the most adjacent word, do
        i. Find the word's node in the graph such that the node's position is respective of any intervening words between the current word and the root phrase.
        ii. If word's node exists in the graph, increment its count
        iii. Else, add a new node to the graph and initialize its count to 1
    b. Repeat the previous step for each word *after* the root phrase to build the second half of the graph.
iv. Weight the graph such that stop words and the root node have no weight and the remaining nodes have weights that are proportional to their frequency.
v. For each path beginning from the root node and continuing until a leaf node, find the path with the greatest total weight. Note whether this path is before or after the root node.
vi. For the path with the greatest weight, create a new graph with a root node that contains all the words in this path.
vii. Filter the original set of posts to only those posts that contain the new root node phrase.
viii. Build a new graph using the same procedure above but only for the opposite direction from the path with greatest weight found earlier. This new graph effectively builds the remaining half of the summary.
ix. Once the new graph is built, find the path with the greatest total weight from the root node to the available leaf nodes.
x. The words along the path with the greatest weight represent the final summary of the PR algorithm.

Figure 4. Phrase Reinforcement Algorithm

Results

As mentioned earlier, the Phrase Reinforcement algorithm produces human comparable summaries for a majority of topics with as few as 100 posts. In order to evaluate the performance of our automated summary generator, we used the following approach. We first gathered a set of testing data by collecting the top ten currently trending topics from Twitter's home page every day for five consecutive days. The topics were all gathered during the evenings at roughly the same time. In addition, we queried Twitter for the maximum number of available posts for each of those trending topics. In all, we gathered fifty trending topics together with approximately 1500 posts for each topic. For each topic, we filtered the set of 1500 posts to remove any spam or duplicate posts from the same user. We also removed any non-English posts. From the remaining posts, we kept the first 100 posts and saved them to a file. We then gave the files for each of the fifty posts to two different human volunteers for generating manual summaries. We also gave the identical set of files to our PR algorithm for generating automatic summaries. We then compared our automated and human-generated summaries.

In order to compare the manual and automated summaries, we adopted one of the metrics from the Document Understanding Conference (DUC) of 2002 that was used to compare automated summaries with manual summaries for a set of documents of varying lengths (Lin and Hovy 2003). In particular, we used their *completeness* metric, which asks a human judge to measure how complete an automated summary expresses the *meaning* of a human generated summary. The measure of completeness is ranked according to five different levels and asks whether the automated summary expresses the meaning of all, most, some, hardly any, or none of the manual summary (Lin and Hovy 2003). We adopted the same approach and asked an independent volunteer to compare the pairs of automatic and manual summaries for each of our fifty testing topics. From the results, we assigned values to the completeness choices that ranged from 5 to 1 for the choices All to None, respective of order. We then averaged the results for the fifty topics. For the fifty topics our volunteer manually scored, our PR algorithm produced an average completeness value of 3.86. Table 2 below gives a sample of the last ten of our testing topics for both the manual and automated summaries and their corresponding score.

Topic	Manual Summary	Automated Summary	Score
#clubrules	#clubrules: What is and is not okay in a dance club.	#clubrules ladies if	2
#iusuallylieabout	Twitter users discuss "what #iusuallylieabout" with each other.	#iusuallylieabout my age	5
Apple Fires Back	Apple fires back at windows 7 with their new commercials	RT Apple Fires Back at Windows 7 in New Ads - '"Apple's "I'm a PC" and "I'm a Mac" dynamic ad duo are at it again i...	5
Balloon Boy	The mother of "Balloon Boy" confesses about the hoax and explains it was her husbands idea.	Balloon Boy Mom Admits to Hoax : According to court records made public today, Mayumi Heene, Ball..	5
Dollhouse	Jonathan Frakes did an amazing job directing the final episode of Dollhouse this season.	Dollhouse is a trending topic, could Fox show it a little more love so it can stay on the air?	3
New Moon	The Trailer for Twilight: New Moon was released.	I just took "Are you a real Twilight New Moon fan?" and got: Real Fan!! Try it:	2
Puck	David Booth of the Florida Panthers, using his hands, picks up the hockey puck and throws it down the ice.	the season...Geno made a great play to keep the puck	3
Saw VI	The movie "Saw VI " receives many positive reviews.	Saw VI is great! My girlfriend started crying and screaming	5
Soupy Sales	The slapstick comedian, Soupy Sales, died at the age of 83 years old.	Soupy Sales Died Today. Who Remembers "White Fang" and "Black Tooth"? RIP	4
Follow Friday	Twitter users are looking for other users to follow for "Follow Friday".	Thanks for the follow friday	3

Table 2. Table comparing our manual (left) and automated (right) summaries for ten testing topics.

Conclusions

Comparing our automated summaries against our human-generated summaries for each of our fifty topics, we began to notice a few patterns of behavior. First of all, the PR algorithm appears to perform best when

a topic has a dominant phrase pattern around the central topic. For example, for the topic "Apple Fires Back" in table 2 above, the topic is naturally part of a larger phrase and results in a good summary. Likewise, for the topic "Soupy Sales", we see that the majority of the people indicate that Soupy Sales "died today" and hence also resulted in a good summary. Whenever a topic is naturally part of a larger phrase, the PR algorithm works well and is able to isolate these dominant phrases from the set of input posts. However, not all topics are surrounded by a common phrase. This is especially true for #hashtag topics which is a convention twitter users have adopted in order to make certain topics easy to find via searching for the hashtag (Hashtag 2009). If the hashtag does not naturally fall within a phrase, then the PR algorithm is not able to generate a dominant phrase around the topic. This is the case with the topic "#clubrules" in table 2. For this topic, most users prepend their post with the hashtag instead of using it as part of a sentence. This can be seen by looking at the first five posts for the topic below in table 3.

#clubrules If u cant dance 2 step yo ass off!!! dont try no crazy shit stay in ya lane ladies and gents
#clubrules: Ladies don't carry big purses 2the club &get mad cause there's no room for u2 dance. Nobody told u to pack an overnight bag smh.
#clubrules ladies once u need 2 spray change shirts and re apply make up its time 4 u 2 go home
#ClubRules Gosh darnit! Tip the waitresses and bartenders! Without that alcohol your night would be lame!
RT @mrslovexlabels: #clubrules ladies; stop pickin up those dollar bills off the floor. Birds.

Table 3. Five posts for the hashtag topic "#clubrules".

Notice that almost every post in table 3 begins with the hashtag "#clubrules" and then varies dramatically afterwards. Since the PR algorithm is looking for common word usage around this topic and there is none, the PR algorithm produces a poor summary. Of course, if a hashtag more naturally can be used as part of a sentence, as is "#iusuallylieabout", then the PR algorithm again is able to produce a fair summary. In the samples that we found, people usually lie about their age.

Finally, we should also note why we weight the graph rather than just use the token frequency exclusively as the weight. Since the PR algorithm attempts to find the path (and hence phrase) with the most weight, it has a natural bias towards longer phrases. These longer phrases do not always represent the most dominant idea or phrase in the graph and therefore need compensating for their length. In order to do so, we initialize a node's weight to its frequency but also penalize it the farther it is from the root phrase. By reducing a node's weight over distance, we found that the PR algorithm becomes more attracted to more central nodes around the root phrase and generates better summaries.

Future Work

Presently, we are working on extending our PR algorithm to providing real-time summaries within specific topics. We are experimenting with using a front-end classifier for producing trending topics within distinct categories such as sports, politics, or world events and then summarizing around these topics in order to generate an automated real-time newspaper.

Acknowledgement

The work performed has been partially supported by NSF grant ARRA: 0851783.

References

Beaumont, C. Mumbai attacks: Twitter and Flickr used to break news, The Telegraph, 27 Nov 2008, http://www.telegraph.co.uk/news/worldnews/asia/india/3530640/Mumbai-attacks-Twitter-and-Flickr-used-to-break-news-Bombay-India.html, accessed 11/21/2009.

Beaumont, C. New York plane crash: Twitter breaks the news, again, 16 Jan 2009, http://www.telegraph.co.uk/technology/twitter/4269765/New-York-plane-crash-Twitter-breaks-the-news-again.html, accessed 11/21/2009.

Bossard, A. and Genereux, M. and Poibeau, T. Description of the lipn systems at tac2008: Summarizing information and opinions, Proceedings of the Text Analysis Conference (TAC), 2008.

Brandow, R., Mitze K., and Rau, L.F. Automatic Condensation of Electronic Publications by Sentence Selection, Information Processing and Management, Volume 31, No 5, pp. 675-685, 1995,

Edmundson, H.P. 1969. New Methods in Automatic Extracting. Technical Report, Department of Computer Science, University of Maryland at College Park, 1969.

Hashtags 2009. http://www.hashtags.org

Hu, M. and Sun, A. and Lim, E.P. Comments-oriented blog summarization by sentence extraction, Proceedings of the sixteenth ACM conference on Conference on Information and Knowledge Management, pp. 901-904, 2007.

Kalita, J.K. Generating Summary Responses to Natural Language Database Queries, M.Sc. Thesis, Department of Computer Science, University of Saskatchewan, 1984.

Kalita, J.K., Colbourn (Jones), M.J. and McCalla, G.I. A Response to the Need for Summary Responses, Proceedings of COLING-84: 10th International Conference on Computational Linguistics, pp. 432-436.

Kalita, J.K., Jones, M.L., and McCalla, G.I., Summarizing Natural Language Database Responses, Computational Linguistics, Volume 12, No. 2, pp. 107-124.

Kalucki, J. Editor. Streaming API Documentation, https://twitterapi.pbworks.com/Streaming-API-Documentation, accessed 11/21/2009.

Kazeniac, A. Social Networks: Facebook Takes Over Top Spot, Twitter Climbs, February 9th, 2009, http://blog.compete.com/2009/02/09/facebook-myspace-twitter-social-network/, Accessed on 11/21/2009.

Kolcz, A., Prabhakarmurthi, V, and Kalita, J. Summarizing as Feature Selection for Text Categorization, Proceedings of the Tenth International Conference on Information and Knowledge Management, CIKM '01, Atlanta, Georgia, pp. 365-370.

Lin, C.Y. and Hovy, E. Automatic evaluation of summaries using n-gram co-occurrence statistics, Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume, pp. 71-78, 2003.

Lin, Y-R, and Sundaram, H. Blog Antenna: Summarization of Personal Blog Temporal Dynamics Based on Self-Similarity Factorization, Proceeding of International Conference on Multimedia and Expo. (ICME 2007), pp. 540-543.

Luhn, P. The Automatic Creation of Literature Abstracts, in IRE National Convention, pp. 60-68, 1958.

Mahesh, K. Hypertext Summary Extraction for Fast Document Browsing, Working Notes of the AAAI Spring Symposium for the World Wide Web, pp. 95-103, 1997.

Schonfeld, E. Pingdom Says People Are Tweeting 27 Million Times A Day, November 12, 2009, http://www.techcrunch.com/2009/11/12/twitter-27-million-tweets-day-pingdo/, accessed 11/21/2009.

Molina, Marcel. Editor. Twitter API Documentation, http://apiwiki.twitter.com/Twitter-API-Documentation, accessed 11/21/2009.

Zhou, L. and Hovy, E. On the summarization of dynamically introduced information: Online discussions and blogs, Proceedings of AAAI-2006 Spring Symposium on Computational Approaches to Analyzing Weblogs, Stanford, CA, 2006.