Summarizing Sporting Events Using Twitter

Jeffrey Nichols, Jalal Mahmud, Clemens Drews
IBM Research – Almaden
650 Harry Rd, San Jose, CA 95120
{jwnichols,jumahmud,cdrews}@us.ibm.com

ABSTRACT
The status updates posted to social networks, such as Twitter and Facebook, contain a myriad of information about what people are doing and watching. During events, such as sports games, many updates are sent describing and expressing opinions about the event. In this paper, we describe an algorithm that generates a journalistic summary of an event using only status updates from Twitter as a source. Temporal cues, such as spikes in the volume of status updates, are used to identify the important moments within an event, and a sentence ranking method is used to extract relevant sentences from the corpus of status updates describing each important moment within an event. We evaluate our algorithm compared to human-generated summaries and the previous best summarization algorithm, and find that the results of our method are superior to the previous algorithm and approach the readability and grammaticality of the human-generated summaries.

Author Keywords
Implicit Crowdsourcing, Journalism, Social Media, Status Updates

ACM Classification Keywords
H.5.2. [Information Interfaces and Presentation]: User Interfaces; H.2.8. [Database Management]: Database applications - Data mining.

General Terms
Algorithms, Experimentation, Human Factors

INTRODUCTION

Everyday, people are posting millions of status updates to social networks, such as Twitter and Facebook. As of June 2011, more than 200 million updates were being posted to Twitter each day [1]. Some of these updates describe events that people are participating in or watching through a media source such as television, including natural disasters [25], political debates [21], and sporting events [11]. These updates provide important information about what is happening in real-time, but this information must be summarized or visualized in order to be accessible to human viewers. If automated summaries could be generated, breaking information could be presented more quickly without waiting for a journalist to manually generate a summary and news stories could be written for smaller events that journalists are not able to cover.

In this paper, we explore an automated method for *implicitly crowdsourcing* summaries of events using only status updates posted to Twitter as a source. We focus on summarizing sporting events, specifically World Cup soccer matches, because each event takes place over a short defined period of time, there is a substantial volume of tweets about each event, and there is press coverage of each event to serve as a gold standard. Specifically we make the following contributions:

- An unsupervised algorithm for generating a textual summary of events, especially sporting events from status updates in Twitter.
- Evaluation of our algorithm's results compared to human-generated summaries using the standard automated ROUGE method [12] and human evaluators that rated summaries in terms of readability and grammaticality. The results of both demonstrations show that our algorithm works reasonably well and outperforms the best-known algorithm for micro-blog summarization [24].

RELATED WORK

A substantial amount of work on text summarization exists in the literature [9]. Unfortunately, techniques developed for summarizing large text documents, such as blogs [30], are not readily applicable for summarizing large sets of status updates, which have very different qualities (e.g., small amounts of text, highly repetitive, and very noisy). Prior work on micro-blog summarization includes summarization of a set of status updates by Sharifi et al. [23, 24] and event summarization by Chakrabarti et al. [7].

Our work leverages a word frequency-based technique, based on the phrase graph technique proposed by Sharifi et al. [23, 24]. Unlike their work, which focused on generating a single sentence from a set of tweets, we broaden the scope to look at an entire event and generate multiple sentences that describe each important moment within an event.

Chakrabarti et al. [7] use Twitter to generate summaries of long running, structure rich events, where multiple events share the same underlying structure. They looked specifically at multiple American football games, and used a modified Hidden Markov Model (HMM) that learns the structure and vocabulary of the event. However, the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
IUI'12, February 14–17, 2012, Lisbon, Portugal.
Copyright 2012 ACM 978-1-4503-1048-2/12/02...$10.00.

assumption of the availability of multiple similar events may not be possible in practice, especially at the granularity of the moments they considered within their events. For example, an HMM is unlikely to capture all possible sequences of sub-events (e.g., interception, touchdown in American football) in an event. Also, certain vocabulary items, such as the names of participants, cannot be learned unless the same two teams have played multiple times, which is not common in some sports or in non-sporting events. Our algorithm is not limited by the availability of multiple similar events. Our approach is also unsupervised, whereas supervision is required to train an HMM model.

Several other systems aggregate tweets on a topic into visual summaries, such as TweetMotif [8], the Visual Backchannel [10], TwitInfo [13], and others [11, 21, 22]. These visualizations must be interpreted by users and do not include sentence-level textual summaries. Although each of these systems has various differences, most label important moments within the Twitter stream with single word labels or word clouds. A common thread across many of these systems is the need to detect important moments within the Twitter stream.

The use of rapid increases (or "spikes") in status update volume over time is a common technique across many systems, and is used in our algorithm as well, however the particular method in which volume is used differs. TwitInfo [13] has an algorithm based on TCP congestion detection that uses the weighted moving average, variance, and the absolute value of volumes to find spikes. Weng et al. [27] describe an event detection algorithm from tweets based on clustering of wavelet-based signals. They use wavelet analysis on the frequency based signals of the words from tweets, filter words which have low signal-auto correlations, and cluster the remaining words to form events with a modularity-based graph partitioning technique. Shamma et al. [21, 22] use a slope-based method similar to ours, but also extend this method to operate on the volume changes in specific topics rather than just the overall volume of recording. Petrovic et al. describe an algorithm that detects events that have not occurred previously [14].

Other systems use domain knowledge or manual supervision to guide their event detection, whereas our algorithm does not rely on any domain knowledge. Hannon et al. use a set of keywords provided by the user along with volume information to determine what events to include in an automatically generated video highlight reel [11]. SportSense (http://www.sportsense.us) uses domain knowledge of American football games (e.g., touchdown, fumble) to visually display the major events that occurred in a game and rate fans' excitement level. Sakaki et al. use a domain specific classifier to detect earthquake events from Twitter status updates [18]. Becker et al. use a supervised classifier to detect events [5]. Popescu et al. detect events from twitter using knowledge of known entities [16, 17].

Our work on detecting and summarizing events using only Twitter data is related to other work that performs the same function using different data sources. For example, Piriou et al. [15] detect events using computer vision from video sources. Baillie et al. [3] use crowd noise from an audio recording to detect important moments within a sporting event, which is analogous to our use of Twitter volume for the same purpose. Xu et al. [28] experimented with both a hybrid audio and video approach to detect moments within a sporting event, and using webcast text to perform the same function. This latter work did not have to consider noise in the text however, because the webcast text was much cleaner than typical Twitter data.

DATASET

For the design and evaluation of our algorithm, we collected a dataset of tweets from 36 games of the 2010 World Cup. Tweets for this dataset were recorded through Twitter's Streaming API using a track stream that receives tweets based on keyword query. Most games were recorded using the keywords "worldcup" and "wc2010" that were promoted by FIFA (www.fifa.com) and Twitter for World Cup games. Note that the Twitter API matches keywords without case-sensitivity and also matches hashtags. Some games occurred simultaneously with others, and for these we used keywords such as the names of the countries, any country abbreviations, and any distinct team names (e.g. "socceroos" for Australia). Due to various technical difficulties, many recorded games contained gaps of 5 minutes or longer where tweets were missed.

Evaluation of the algorithm, as you will see, requires substantial human effort to identify important moments, create manual summaries, and rate automated summaries, so we reduced the set of 36 games to just 3. These 3 games can be seen in Table 1, and were selected randomly from the set of games with complete recordings.

Game	Tweets	Max Tweets/Min
US vs. Slovenia	113189	3992
Germany vs. Serbia	72335	1810
Australia vs. Serbia	77517	8235

Table 1. Summary of World Cup Dataset

SUMMARIZING SPORTING EVENTS

Sporting events consist of a sequence of moments, each of which may contain actions by players, the referee, the fans, etc. Our algorithm relies on Twitter users to collectively identify the important moments within an event and also describe those moments. At a high level, our algorithm relies on two properties of the Twitter stream:

- Sudden increases, or "spikes," in the volume of tweets in the stream suggest that something important just happened because many people found the need to comment on it.

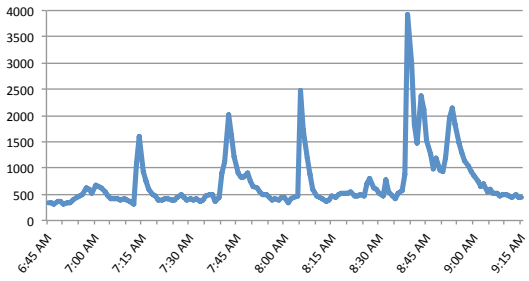

Figure 1. Twitter volume graph for the 2010 World Cup game of US vs. Slovenia. The x-axis is time and the y-axis is volume as measured in tweets/minute.

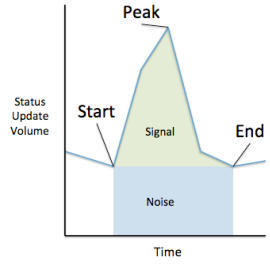

Figure 2. The anatomy of a spike, showing start, peak, and end times.

- A high percentage of the contents of the tweets at a "spike" in volume contain text describing what happened in the moment, and this text often contains repetitive elements, such as the names of players involved, the type of the event, etc.

Our algorithm breaks the problem of generating a summary of an event into the following steps, which are described in more detail in the following sub-sections:

- Status updates are collected using basic keyword filtering.
- The important moments within an event are detected by searching for extreme changes in update volume on a per minute basis.
- Status updates from each important moment are identified.
- Several noise reduction algorithms are used to eliminate spam and off-topic status updates.
- A word frequency table is calculated using the longest sentence in each status update.
- Sentences from the status updates in each cluster are ranked using the phrase graph, and the top *n* sentences containing no overlapping tokens are returned.

Detecting Important Moments

Shamma et al. [21, 22] and Marcus et al. [13] have noted that important moments can often be detected in Twitter streams when the volume of status updates increases sharply. Over the course of a sporting event, this may happen many times (for example, see Figure 1).

We could have used a moment detection algorithm that relies on the absolute value of the volume, however we discovered two problems with this approach. First, sometimes the stream volume may stay high for several minutes and have multiple localized peaks. Second, some moments generate significantly less traffic than others and might be missed by a technique that relies on absolute values. To avoid the above problems, we chose to use a detection algorithm based on the change in volume (i.e. the slope of the volume graph). In this paper, we measure tweet volume at the granularity of tweets/minute.

Our algorithm is motivated by spike detection as described in [21]. In particular, we have implemented both an offline version and an online version of our algorithm for moment detection. When our algorithm is used offline (i.e. tweets from the entire event are available), it computes a threshold for the entire event from basic statistics of the set of all slopes for that event. For example, a threshold for a particular soccer match may be computed from the tweets recorded for that entire match. In the online approach, we compute this threshold from the tweets in a moving window with start time and end time, where the size of the moving window is a parameter. The experiments reported in this paper use the offline approach to detect this threshold from the entire event.

We tried several different formulas for computing the slope threshold. The first threshold we tried was median + 2 * standard deviation, in order to include only extreme outliers in the set. Unfortunately, we found that for events with very large spikes the standard deviation in the slope is too large to find the smaller but still meaningful spikes. After some experimentation, we settled on a threshold of 3 * median, which produced results that closely matched a visual inspection of spikes across our larger 36 game data set. In future work, we plan to examine this heuristic to see how it applies to other types of sporting events not in our data set.

After identifying all slopes that exceed the threshold, we generate a list of "spikes" that correspond to the important moments in the event. Each spike can be defined as the tuple of <Start Time, Peak Time, End Time>. For each slope above threshold, we calculate the start time by searching backwards in time until we find the point where the slope began going up. The peak time is calculated by searching forward until we find the point where the slopes start decreasing. Finally, we calculate the end time by searching forward from the peak time to find the point where the slope begins increasing again (see Figure 2). Before returning the list of spikes, we remove any duplicates, which may happen for large spikes that include multiple above threshold slopes.

Selecting Updates for Moments

Given an important moment, as described by a spike tuple, we need to select a set of status updates that are representative of that moment. We could, for example, choose to select all status updates sent between the start and

end time of the spike, but we have found in practice that this option is not desirable. This is because there is often a relatively constant amount of noise throughout the stream, whereas the "signal" is greatest at the peak and goes to zero at the start and end of the spike (see Figure 2). Thus, our goal is to select the largest possible set of status updates within the spike that also have a high signal-to-noise (SNR) ratio. This is a multi-variable optimization problem [26].

We use a heuristic approach to select a time range that optimizes both SNR and G, the number of signal tweets that can be extracted. This time range can be calculated for a time interval given knowledge of the noise over that interval. We approximate the noise over a time interval as the tweets arriving per unit time (e.g., minutes, as in our implementation) at the time the spike begins or ends (see Figure 2) multiplied by the length of the time interval. We start with the peak time T_p, calculate SNR and G for time intervals (T_p, $T_p + \Delta t$), ($T_p - \Delta t$, T_p) and ($T_p - \Delta t$, $T_p + \Delta t$), and iteratively increase the value of Δt (1 min, 2 min, etc.). We select a time interval where the SNR for that interval is above a threshold θ_{SNR} and the ratio of the number of signal tweets within the interval to the total signal tweets over the entire spike is above a threshold θ_G. If there are multiple such intervals, we select the one that yields the most status updates. The thresholds are set experimentally, and we found that $\theta_{SNR} = 0.7$ and $\theta_G = 0.6$ work reasonably well for our dataset.

Noise Elimination

Once we detect important moments and the status updates corresponding to those moments using the previous step, we use several noise elimination techniques to filter out spam and off-topic tweets. First, we use a language detector from the Apache Nutch project (http://nutch.apache.org/) in combination with the language information reported by Twitter to detect the languages of each update and remove those that are non-English. Note that the information provided by Twitter is unreliable and often reports that English is the language of a tweet when it is not, necessitating the use of another language detector. Second, we use a set of heuristics to filter out spam and types of tweets that we have generally found to not be useful. Spam is filtered using a dictionary of common terms that we found in spam tweets. Other heuristics include removing tweets that are replies to other users and tweets that contain URLs. In general, our algorithm will ignore most spam anyway, provided the signal from the tweets in the spike dominates the noise (generally true). Next, we perform a normalization process where we eliminate repeating characters from a term, e.g., goooooooal becomes goal as a result of normalization.

Extracting Summary Sentences

The final step is to find the N sentences contained in our set of status updates that best summarize the set. The set of updates could either come directly from the noise elimination step or from each of the clusters created in the clustering step. Sentences are extracted from tweets using heuristics, primarily by splitting on punctuation symbols. Our approach is to construct a phrase graph from the longest sentence in each status update, weight the graph according to frequency of words and a few other heuristics, and then to score the longest sentence using the phrase graph. The use of a phrase graph and the weighting scheme is borrowed from Sharifi [23, 24], however unlike their technique that only generated a single sentence per topic, we generate multiple sentences that describe each of the important moments within an event. In addition, our approach incorporates several important differences from Sharifi's algorithm that we will describe in detail.

The intuition for using a phrase graph is that many words and short phrases are repeated by status update authors, either intentionally through mechanisms such as Twitter's ReTweet feature or unintentionally because there are common elements in the moment to which authors refer (e.g., "goal" or the name of a key player). The phrase graph consists of a node for each word appearing in any status update, and an edge between each set of two words that are used adjacently in any status update (see Figure 3).

Repetition is represented through weights assigned to the nodes, and optionally the edges, that reflect the frequency with which the word occurs in the set of status updates. We have experimented with several methods of weighting, described later, and settled on the basic approach of using a weight equal to just the number of occurrences of the word. The weight of stop words and hashtags are set to 0. For example, Figure 3 shows the phrase graph that would be generated from these status updates: *Landon Donovan scores!*, *Donovan scores a brilliant strike*, *Landon Donovan is brilliant*.

We construct the phrase graph from only the longest sentence in each status update, rather than using the entire update. This helps to ensure that our generated summaries are well formed and eliminates some of the noise found in updates. We also remove hashtags from the beginning and end of each extracted sentence, which often are meta-data descriptors for the update rather than meaningful language that should be included in the sentence. The language is still often noisy, and we may consider other techniques for cleaning the sentences in future work. Note that we did not include a virtual END symbol in our phrase graph because we are scoring existing sentences rather than generating new ones.

Sharifi's algorithm uses a phrase graph to generate a single

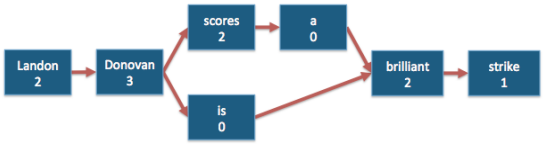

Figure 3. An example phrase graph. Words and their respective weights are shown in each box.

sentence summary [23]. In contrast, we leverage the phrase graph to score the input sentences, each extracted from one status update, and find the best scoring sentences. To score a sentence, we tokenize the sentence, remove any duplicate tokens, and sum the weights of all nodes matching a token to generate a score. Duplicate tokens are removed so that tokens that appear multiple times only contribute to the score once. We also tried a variant of this algorithm that included edge weights in the score, but we found that node weights alone were sufficient to generate good summaries. We also considered normalizing the score by the number of tokens in the sentence, however normalized scores seem to favor shorter sentences made up of a few high scoring terms whereas not normalizing favors longer sentences with more information.

Given the list of scored sentences, we then output the top N sentences that *do not share any non-stop word stemmed tokens*. This ensures that the sentences each include different information, although this is sensitive to the quality of the stemming algorithm (we use the Porter stemmer [18]) and the tweet normalization applied during noise elimination. The value N may be chosen based on the summarization application; we find that choosing N=3 seems to work well.

EVALUATION

Here we describe our evaluation metrics, experimental setup, and our quantitative and qualitative results.

Evaluation of Important Moment Detection

To evaluate the performance of our moment detection algorithm, we compare the counts of various types of key moments detected by our algorithm to counts extracted from reading several recap articles about each game. We chose the following categories of key events in a soccer game: goals, penalties, red cards, yellow cards, disallowed goals, game start, game end and half time.

To establish event counts for our algorithm, one author manually examined all of the status updates selected to represent each important moment and identified the number and type of the events described in that moment. Note that an important moment may contain multiple key events, e.g., a red card followed by a penalty, or a goal followed by half time. An important moment may not contain a reference to any key event, and we consider these "noisy" moments.

To establish actual event counts for these games, one author examined game recap articles from four sources: FIFA.com, ESPN.com, New York Times and Yahoo! Sports. The first two sources were also used for our summarization evaluation later. We used multiple sources to ensure that no key events were missed.

Recall (R) of moment detection is defined as the ratio of the number of key events detected by our algorithm and the total number of key events in the entire game. Precision (P) has the same numerator, but the denominator is the sum of the total number of key events detected by our algorithm plus the number of "noisy" moments.

Game	Actual Events	Detected		R	P
		Moments	Events		
US vs. Slovenia	13	9	8	0.62	0.89
Germany vs. Serbia	16	8	11	0.69	0.92
Australia vs. Serbia	11	9	10	0.91	0.91

Table 2. Recall/Precision Performance of Moment Detection

Table 2 shows overall performance in terms of precision and recall. Observe that precision performance is very high for each game, which means that when our algorithm detects an important moment it likely contains at least one key event.

Recall performance is somewhat low for the first two games. To further examine this, we computed recall for each key event type. Table 3 shows that our algorithm can accurately identify goals, red cards, penalties, game ends, and disallowed goals. This makes sense, because these key events cause significantly large spikes in tweet volume during these games. However, yellow cards, game starts, and half times did not cause sufficient increases in tweet volume in our data set and are thus harder to detect. Note that detection of game ends was better than game start events. We have observed that people often tweet their reactions to the game immediately after the game ends. For example, at the end of the US vs. Slovenia game there are many complaints about the referee and a disallowed goal.

Key Event Type	Recall
Goal	1.0
Red Card	1.0
Yellow Card	0.53
Penalty	1.0
Game Start	0.67
Game End	1.0
Half Time	0.33
Disallowed Goals	1.0

Table 3. Recall of Key Event Detection

Evaluation of Summarization

For the second experiment, we use two methods:

- The ROUGE-N metric to compare our summaries to gold standards generated by humans [12].
- Human evaluation of the summaries for readability, grammatical correctness and meaning in comparison to the gold standards.

Game	Spike	Manual Summary	Our Summary	TF-IDF Summary
US vs. Slovenia	1	In the first 15 mins of the soccer game between USA and Slovenia, Slovenia is leading with a goal by Birsa. Birsa scored an easy goal from midfield to the right of the goal, as USA left that shot wide open. Terrible defense by USA team, too much space left open.	Good goal for Slovenia and the USA once again starts a game terrible. Birsa gives #SVN 1-0 lead with smart shot. Howard didn't even look like he saw that one coming.	The Slovenians strike first, #USA standing around waiting for a whistle, or a bus, or something. vs #SVN 0-1 '15 GOAL by Birsa #USMNT defende looked hipnotized, Howard questionable positioning was a factor too. #usa playing like high skool kids in a practice match with no game plan.
Germany vs. Serbia	3	Klose argues with referee, gets second yellow cards and is out of the game. Germany down to 10 men. 1-0 Serbia.	Germany screwed by the refs and a red card for Klose; seconds later, a pretty goal by the Serbs. yellow seems to be a very popular colour in this game.	So first I didn't make my bet on a red card, then Germany conceed meaning I lose points for TWO defenders in my fantasy team. This Ref is absolutely insane, just 35 mins in, 7 bookings & already #GER star Miroslav Klose is sent off on a bullshit challenge. Mick McCarthy claiming the referee is 'a clown' = best line of the world cup so far.
Australia vs. Serbia	9	Serbia Australia match ends with 2-1. With a result of 1:0 between Germany and Ghana this means that Ghana and Germany will advance to the knock out rounds and serbia and australia will be out.	Australia won 2-1 on serbia, Germany won 1-0 vs Ghana, Germany and Ghana goes on to the next round. Great win by #aus but not good enough to go through. Final score #Aus 2 #Srb 1.	Kalo timnas kita pake pelatih luar spt AUS & GHA ini, apkh kita bs masuk Piala Dunia. Germany beats Ghana, Australia beats Serbia, which means, Germany and Ghana continue on, and Ghana will face the United States. Socceroos leading 2-1, Gillard leading Rudd, crowds on either sides of earth gather with vuvuzelas.

Table 4. Sample summaries for each game in our data set

We use two gold standards:

- Game "recap" articles written by FIFA.com and ESPN.com.
- Manual summaries generated by humans from our Twitter corpus for each important moment detected by our algorithm.

The recap articles allow us to evaluate our overall summary of the game, although these articles are longer than the summaries generated by our algorithm. The manual summaries generated by humans for each spike allow us to evaluate our summaries of each important moment separately. To generate the manual summaries, we recruited six humans and randomly assigned each to summarize one game (2 subjects for each game). Each subject was given a list of tweets for each moment as identified by our algorithm and asked to generate summaries for each moment of at most three sentences using only words that appeared in the tweets. Our subjects were busy researchers from our lab, so to keep the task manageable we provided them with a randomly generated subset of at most 750 tweets for each moment and limited them to an hour to generate all summaries. Pilot studies showed that subjects were able to generate high quality and representative summaries given these constraints.

We chose the ROUGE-N method to compare summaries because it is widely used in the summarization community, it is automated, and its results have been shown to correlate strongly with human judgments of summary quality [12]. ROUGE-1, which compares the similarity of summaries at the level of unigrams, has been shown to work well for short summaries, such as headlines, and used for evaluating tweet-sized summaries [24]. ROUGE-2, which uses bigrams, has been shown to work well for longer summaries. We use ROUGE-1 to compare our summaries of important moments, because they are relatively short, and ROUGE-2 to compare the full game summary with the recap articles. The automated nature of ROUGE allows us to easily compare many different variations of our algorithm. We also compare our results to the modified TF-IDF algorithm of Sharifi et al. [24], which is the best tweet summarization algorithm of which we are aware.

Automated summary evaluations are not perfect however, and so we conducted a human evaluation of the automated summaries generated by the best variants of our algorithm and Sharifi's modified TF-IDF algorithm. Each summary was evaluated by 3 humans. They were shown both the automatically generated summary and the human-generated summaries for each important moment. Each evaluator then used a 7-point Likert scale to rank the automatically generated summaries on three dimensions: readability, grammatical correctness, and inclusion of content found in the manual summaries. To provide a baseline and to assess the quality of the manual summaries, we also asked the evaluators to rate the readability and grammatical correctness of the manual summaries.

Automated Summary Evaluation Results

Table 4 shows a selection of summaries generated by our algorithm and the modified TF-IDF algorithm. We start by examining the performance of our algorithm relative to the manual summaries generated for each important moment in a game. This evaluation compares our summary generation to Sharifi's modified TF-IDF algorithm and a human generated gold standard. Figure 4 shows the results.

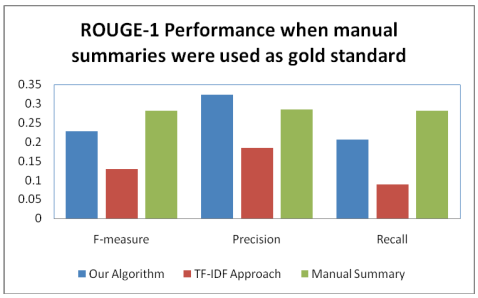

Figure 4. ROUGE-1 comparison of our algorithm, modified TF-IDF algorithm and manual summaries for important moments within an event.

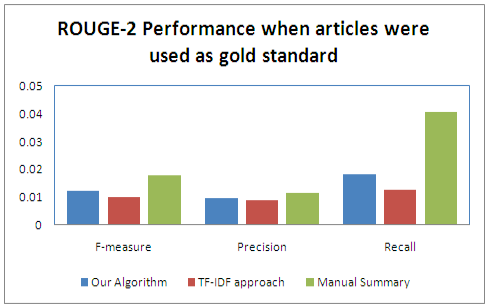

Figure 5. ROUGE-2 comparison of our algorithm, modified TF-IDF, and the manual summaries using recap articles as the gold standard.

These results show that there is fairly large variation between the two manual summaries for each game, which is reflected in the relatively low numbers for their comparison using ROUGE-1. Our algorithm performs just below this high bar, and even achieves a higher precision than the manual summary comparison. This indicates that our summaries at least occasionally must contain elements of both manual summaries that are not shared between the manual summaries. We find no significant difference between the summaries generated by our algorithm and the manual summaries ($p \sim= 0.684$, paired t-test, n=26). Our algorithm significantly outperforms the modified TF-IDF algorithm ($p < 0.001$, paired t-test, n=26).

The summaries for each important moment can then be concatenated together to form a full game summary. To evaluate the quality of these summaries, we compare them to press articles using the ROUGE-2 algorithm. Figure 5 shows the results. Note that the manual summary evaluation is different for articles than it was for moments. Here we concatenate the manual summaries generated for important moments into a full game summary, and use ROUGE-2 to compare with the two recap articles. As before, the numbers for the manual summaries represent an upper bound of what we expect our automated algorithms to accomplish. Note that the manual summaries and all algorithms have low performance on the ROUGE-2 test, largely because the recap articles are substantially longer than the concatenated summaries. There were also mismatches in word use, perhaps because journalists typically use different words than Twitter users. The recap articles also contain post-game quotes from players and coaches, which are not contained in the datasets available to our summarization algorithms. Still, we find it instructive that our algorithm appears to perform better than the modified TF-IDF approach, although n is too small (n=3) to test for statistical significance.

	Mean	Median	Std. Dev.
Readability	6.01	6	1.00
Grammaticality	5.60	6	1.19
Content	5.19	5	1.39

Table 5. Results from 3 human evaluators for our algorithm using a 7-point Likert scale

	Mean	Median	Std. Dev.
Readability	3.75	3.66	1.62
Grammaticality	3.8586	3.5	1.7872
Content	3.5754	3.3333	1.8763

Table 6. Results from 3 human evaluators for TF-IDF algorithm using a 7 point Likert scale

	Mean	Median	Std. Dev.
Readability	6.53	7	0.59
Grammaticality	6.37	6.67	0.73

Table 7. Results from 3 human evaluators for manual summaries using a 7-point Likert scale

Human Evaluation Results

Results of the human evaluation of the summaries are shown in Table 5, Table 6, and Table 7. Note that ratings for inclusion of content from the manual summaries are only available for the algorithm-generated summaries. These results suggest that our human evaluators found the summaries generated by our algorithm to be understandable and contain much of the same information as the manual summaries, thus confirming the automated evaluation. The similar readability and grammaticality ratings for our results compared to the manual summaries suggest that our summaries may be of suitable quality for human consumption.

ALGORITHM VARIANTS

During our algorithm design process, we considered several variants to our algorithm. These included the use of a keyword filtering method, clustering, and several weighting schemes for the phrase graph. In this section, we describe these variants and the results of experiments comparing these variants to our final algorithm. We include these negative results to assist future researchers in creating new tweet summarization algorithms.

Keyword Filtering

During the noise elimination phase of our algorithm, we considered using a keyword filtering approach to eliminate off-topic tweets using a two-stage process. This is conceptually similar to the centroid-based content selection

method proposed by Becker et al. [6]. We start by identifying the top-M keywords for each important moment using TF-IDF [20]. Here the value M is determined experimentally. For computing IDF required by the TF-IDF approach, we use a different approach for offline and online summarization. If we have access to all of the status updates for an event (offline), we build a corpus using status updates from every important moment. In the online case, we use a corpus of status updates collected from important moments found in previous similar events, e.g., other World Cup games. In either case, each moment corresponds to a Document and IDF is calculated across moments for a given match. Keywords identified using this approach represent those that are unique to the current important moment but do not appear evenly across all moments. We eliminate all status updates that do not contain any of these keywords under the assumption that these updates are likely off-topic.

To assess the effect of keyword filtering, we tested both our summarization algorithm and Sharifi's TF-IDF algorithm with and without this feature. We observe that the performance of our algorithm drops with keyword filtering (about a 5% decrease on average for both summary types), perhaps because some of the tweets removed for being off-topic do contain topical information that positively affects the phrase graph weighting. Keyword filtering improves the performance of the modified TF-IDF algorithm however (about a 50% increase on average for both summary types). Perhaps the TF-IDF technique's topically important term selection is sensitive to noise and filtering removes some of that noise. The results presented in the Evaluation section were computed using the best choice for each algorithm; filtering was turned off for our algorithm, but was turned on for the modified TF-IDF algorithm.

Clustering

Clustering would seem to be an intuitive method for sets of tweets describing different aspects of an important moment. For example, in a soccer game a goal moment might be described both in terms of the goal scorer (e.g., brilliant strike to the top of the net) and the goalkeeper's reaction (e.g., caught flatfooted).

We tried a number of clustering algorithms, such as EM and KMEANS from WEKA [2], but through experimentation we observed that none of these methods were more effective than our non-overlapping token sentence selection method. At the important moment level using manual summaries as the gold standard, we observed approximately a 10-15% performance drop when clustering was used. When articles were used as the gold standard, we observed a performance increase of 5-10%, though this may be meaningless given the overall low performance. A visual inspection of the clusters suggests that the algorithms we tried do a poor job of creating clusters that are topically different. This leads to similar updates existing in multiple clusters, which results in a final summary that includes several similar sentences. In contrast, our non-overlapping token method relies on selecting sentences from the scored list that are demonstrably different because they do not share any keywords. This latter option guarantees diversity, which seems to improve the final result.

Different Sentence Scoring Approaches

To determine the best sentence scoring method using the phrase graph, we varied different factors such as a penalty for distance from the term with maximum frequency [23] and the inclusion of edge weights. We tried two edge-weighting options: weights on all edges based on frequency and weights only on edges between non-stop words. We computed summarization performance for each of those cases.

We found the penalty for distance from the term with maximum frequency hurt performance in all cases, which differs from the Sharifi's results. This may be because Sharifi's work focused on generating summaries for tweets selected based on a single keyword, which is not true for our dataset.

We also found that the weighting scheme that did not use any edge-weights performed the best, though only slightly better than version that used edge weights on non-stop word edges. Thus the best variant is, unfortunately, the same as basic frequency scoring, which would not require the creation of the phrase graph. While it is disappointing that the phrase graph does not significantly improve the results of summarization, it is useful to note that simple approaches can yield quality results. The very close performance with and without non-stop word edge weights also suggests that there may be ways to use of the phrase graph in more beneficial ways, and we plan to examine this in future work.

No.	Add Penalty?	Add Edge Weight?	F
1	Yes	Yes	0.2058
2	Yes	No	0.21111
3	Yes	No stop word weights	0.2107
4	No	Yes	0.2198
5	No	No	0.2274
6	No	No stop word weights	0.2273

Table 8. Performance comparison for different sentence scoring options using the phrase graph

APPLICATION TO OTHER SPORTING EVENTS

We have presented the evaluation of our summarization algorithm for World Cup soccer matches, however we have also experimented with applying our algorithm to other types of sporting events, such as baseball. We collected tweets for two major league baseball games during the summer of 2011: Boston vs. Seattle and Boston vs. Kansas. Tweets were recorded using a similar method with the keywords "baseball," "mlb," the names of the teams, the

Spike 7: Boston Red Sox vs. Seattle Mariners
Josh reddick homerun puts the red sox up 5-4 on seattle. #redsox take a 5 to 4 lead and put Lackey in position to win despite sucking. You hear all those cheers following # red sox

Table 9. Sample summary for a spike from a baseball game

cities the teams were from, and the last names of the starting pitchers.

When we applied our algorithm to detect important moments from those games, it detected 11 moments from the first game and 9 moments from the second game. We conducted a similar important moment evaluation as before, using baseball-specific events such as run, home run, single, double, stolen base, hit, game start, game end, and inning break. For manual annotation, we used game recap articles from ESPN. We found that our algorithm could detect key events such as game start (100% recall), game end (100% recall) and home runs (approximately 75% recall). Those key events are typically associated with large number of tweets. Our algorithm also detected some runs (approximately 60% recall), however it showed poor performance (20-25% recall) for detecting detailed events such as singles, doubles, and inning breaks. These events typically contain lower tweet volumes than home runs or game start and hence our moment detection algorithm fails to detect them. In the future, we could use domain knowledge such as use of certain keywords to search tweets for these key events, perhaps employing a topic method similar to Shamma et al. [22]. We also examined precision and found that 2/11 moments from the first game and 1/9 moments did not correspond to any key event.

We also examined the summaries generated by our algorithm for important moment (see Table 9 for an example). Summary quality appeared to be comparable to the World Cup results, provided that the moment being summarized referenced some key event.

CONCLUSION AND FUTURE WORK

Our summarization algorithm seems to produce reasonable sentence-level summaries of important moments, and then these moment summaries can be concatenated to produce an event summary of a few paragraphs. We believe these event summaries could communicate what happened during an event to a person who could not follow it in real-time.

One question we have not addressed is the inclusion of bias in our generated summaries. For a sporting event, status updates often include an opinion in favor or against one of the teams that is competing, and we would like to take this bias into account. One option would be to generate summaries from only neutral status updates. Alternately, it might be desirable to include updates with just a single bias. Developing automated methods of detecting bias would be key to this work.

We would also like to improve the quality of the generated summary sentences. Our approach produces reasonable sentences from tweets, however we are curious to try other approaches, such as lattice-based paraphrase generation [4].

We have presented the evaluation of our algorithm in detail for soccer matches, with a brief discussion of an evaluation with baseball games. A key question is how our algorithm might apply to different, possibly longer term, events. We believe it should be possible to apply our technique to longer-term events because projects, such as TwitInfo [13], have used techniques similar to ours to detect sub-events as part of visualizing longer-term events. One feature that might be needed for other event types is the inclusion of domain information. In this paper, we have explicitly avoided using domain information about sports and soccer in particular, but this may be necessary for some events. For example, events in baseball are sometimes dependent on each other, and detecting the manner in which a run scored may require knowledge of baseball to extract a reasonable summary of a run-scoring event.

To support smaller scale events, which would be needed to generate journalistic descriptions of events that journalists do not cover, we will need to understand what the signal/noise and volume requirements are for our algorithm. It seems that our algorithm generates summaries with less consistent quality for smaller spikes, where the signal/noise ratio is worse than for larger spikes. Our algorithm also depends on repetition in status updates, and it is unclear what volume will be needed to get sufficient repetition to generate a reasonable summary. We plan to explore these issues in future work.

We believe this work represents a unique step towards crowdsourcing journalism. Unlike previous attempts, which have explicitly recruited participants, our method *implicitly crowdsources* a journalistic description from information that users are posting for their own reasons. We are excited to explore how this technique might change the way journalists do their work.

REFERENCES

1. http://blog.twitter.com/2011/06/200-million-tweets-per-day.html
2. WEKA, http://www.cs.waikato.ac.nz/ml/weka/
3. Baillie, M., Jose, J. M. An Audio-based Sports Video Segmentation and Event Detection Algorithm, In *Proc. CVPRW'04*.
4. Barzilay, R. and Lee, L. Learning to Paraphrase: An Unsupervised Approach Using Multiple-Sequence Alignment, In *Proc. HLT-NAACL 2003*.
5. Becker, H., Naaman, M. and Gravano, L. Beyond Trending Topics: Real-World Event Identification on Twitter. In *Proc. ICWSM 2011*.
6. Becker, H., Naaman, M. and Gravano, L. Selecting Quality Twitter Content for Events. In *Proc. ICWSM 2011*.

7. Chakrabarti, D. , Punera, K, Event Summarization using Tweets, In *Proc. ICWSM 2011*.
8. Connor, B. O'., Krieger, M., Ahn, D. TweetMotif: Exploratory Search and Topic Summarization for Twitter. In *Proc. ICWSM 2010.*
9. Das, D., Martins, A.F. T. A survey of automatic text summarization, CMU technical report. http://www.cs.cmu.edu/~nasmith/LS2/das-martins.07.pdf
10. Dörk, M., Gruen, D., Williamson, C., and Carpendale, S. A. Visual Backchannel for Large-Scale Events, *IEEE Transactions on Visualizatoin and Computer Graphics*, 16(6):1129-38, 2010.
11. Hannon, J., McCarthy, K., Lynch, J., Smyth, B. Personalized and Automatic Social Summarization of Events in Video, In *Proc. IUI 2011*.
12. Lin, C. ROUGE: A Package for Automatic Evaluation of Summaries. Prof. of the Workshop on Text Summarization Branches Out, post conference workshop of *ACL 2004*.
13. Marcus, A., Bernstein, M. S., Badar, O., Karger, D. R., Madden, S., Miller, R. C., TwitInfo: Aggregating and Visualizing Microblogs for Event Exploration. In *Proc. of CHI 2011*.
14. Petrovi´c, S., Osborne, M., and Lavrenko, V., Streaming first story detection with application to twitter. In *Proc. of NAACL 2010*.
15. Piriou, G., Bouthemy, P., Yao, J. Learned probabilistic image motion models for event detection in videos, In *Proc. ICPR 2004*.
16. Popescu, A., Pennacchiotti, M., Paranipe, D., Extracting events and event descriptions from Twitter, In *Proc. of WWW 2011.*
17. Popescu, A. , Pennacchiotti, M., Detecting Controversal Events from Twitter, In *Proc. of CIKM, 2010*.
18. Porter, M.F., An Algorithm for Suffix Stripping, *Program*, 14(3): 130–137, 1980
19. Sakaki, T., Okazaki, M., Matsuo, Y. Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors, In *Proc. WWW 2010*.
20. Salton, G., Wong, A. , Yang, C.S. A vector space model for automatic indexing. *Comm. of the ACM*, 18(11): 613–620, 1975.
21. Shamma, D. A., Kennedy, L., and Churchill, E. F. Tweet the debates: Understanding Community Annotation of Uncollected Sources, In *Proc.* WSM 2009.
22. Shamma, D.A., Kennedy, L., Churchill, E.F. Peaks and persistence: modeling the shape of microblog conversations. In *Proc. of CSCW 2011*.
23. Sharifi, B., Hunton, M.A.,and Kalita, J. Summarizing Microblogs Automatically: In *Proc. ACL-HLT 2010*.
24. Sharifi, B., Hunton, M.A., and Kalita, J. Experiments in Microblog Summarization. *IEEE Second International Conference on Social Computing, 2010*.
25. Starbird, K., Palen, L., Hughes, A., and Vieweg, S. Chatter on The Red: What Hazards Threat Reveals about the Social Life of Microblogged Information. In *Proc. of CSCW 2010*.
26. Steuer, R.E., *Multiple Criteria Optimization: Theory, Computations, and Application*. New York: John Wiley & Sons, Inc., 1986
27. Weng, J., Lee, F. Event Detection in Twitter, In *Proc. of ICWSM 2011*.
28. Xu, M., Duan, L.Y., Xu, C.S., Tian,Q., Fusion Scheme of Visual and Auditory Modalities For Event Detection In Sports Video, In. *Proc.* of *ICME 2003*.
29. Xu, C., Zhang, Y.F., Zhu, G., Rui, Y., Lu, H., and Huang, Q. Using Webcast Text for Semantic Event Detection in Broadcast Sports Video, *IEEE Transactions on Multimedia,* Vol 10 No 7, Nov, 2008.
30. Zhou, L., and Hovy, E. On the summarization of dynamically introduced information: Online discussions and logs, In *AAAI-2006 Spring Symposium on Computational Approaches to Analyzing Weblogs*.