Summarizing Events using Twitter Updates

Vu Ngoc Son Hitoshi Nishikawa Takenobu Tokunaga

Department of Computer Science, Tokyo Institute of Technology

vu.n.aa@m.titech.ac.jp {hitoshi, take}@cs.titech.ac.jp

1 Introduction

Popularity of social networks such as Facebook® and Twitter® has grown rapidly in recent years. With millions of users posting status updates everyday, these platforms have become a valuable source of information. Users' updates often contain description, opinion about events happening around them in real-life. However, as the amount of updates for such events is usually very large, for instance, hundreds of thousands of tweets, some sort of summarization is required. If such a summary is available, it is possible for users to access latest information without having to wait for other human-generated sources such as newspaper, etc.

The objective of this research is to develop a method for generating summaries of events using Twitter status updates. We also limit our summarization target to events which are long-running and consist of multiple smaller events (sub-event) such as sports events, live broadcast, etc. In our experiment, we use Twitter updates of soccer matches as the input source for summary generation. Output summaries are evaluated by comparing with news reports of corresponding matches.

2 Related Work

There has been research on microblog summarization using various approaches.

Sharifi et al. [7] proposed a phrase graph technique to create a summary from a set of input tweets. However, this algorithm only generates one sentence as the final summary, which is as short as a single tweet. Therefore this method is not suitable for our summarization target.

Takamura et al. [8] viewed the task of summarizing short documents on a timeline as a facility location problem with a linear-partition constraint. Chakrabarti et al. [1] used a modified Hidden Markov Model to segment the timeline into sub-events and then produced summaries for those sub-events. This method required the system to be trained, and it is thus limited to events in which multiple similar sub-events are available.

A common characteristic among long-running events which contains multiple important moments (sub-events) is burstiness [2]. Bursts occurring, usually shown as increase in certain tweet statistics such as tweet frequency, can be viewed as a signal for important moments in an event. There are several studies in which burst detection is used to better identify and extract important information which can improve quality of output summaries.

Nichols et al. [5] used a method in which they first detect and extract bursts from a tweet stream and then select summary tweets for each of them. For sentence extraction they adapted Sharifi's phrase graph based method. However, unlike Sharifi's method, it's possible to select multiple sentences for each sub-event.

Kubo et al. [3] also used burst detection in order to extract important moments. Their sentence extraction method combined tweet scoring with user scoring. Their scoring method emphasized how often a user posts explanatory tweets. Since only one tweet is selected as a representative for each sub-event, it is difficult to deal with cases in which multiple important moments occurs during the same burst period, e.g. a red card followed by a penalty.

In our research, we also make use of burst detection to help extract important moments from the stream.

3 Summarization Method

3.1 Observations

First, by monitoring Twitter stream, we observed the following properties:

- The stream often contains *bursts*, which can be defined as sudden increase of tweet volume. They usually coincides with important moments of a match such as goal, red card, etc. This can be explained as such moments encourage users to make comments or to report about what happened, resulting in increase of user activity.

- During burst period, there is a large amount of similar tweets describing what happened.

Therefore, by focusing on detecting bursts, important information can be extracted more effectively.

3.2 Method

In order to apply burst detection to the summarization process, we produce a summary through two steps. We first detect burst periods among a tweet stream, and then select important tweets by making use of detected burst periods.

3.2.1 Burst Detection

We perform burst detection based on monitoring tweet volume over time. We first calculate the number of tweets posted every minute and then judge a burst to be occurred when the tweet volume exceeds a certain burst threshold. As a result we obtain a list of bursts, each burst is defined as a pair of time points $\{start\ time, end\ time\}$. This makes it convenient to extract tweets in a burst period as well as to evaluate our burst detecting algorithm.

3.2.2 Summary Generation

To evaluate the effectiveness of burst detection in generating a summary for a tweet stream, we compared two extractive method for summary generation.

- **Burst Period as Segment (BPS)**: Tweets are extracted only from burst periods and used as input for our summarizer. This method can reduce calculation time drastically, however there is a risk of losing important updates if the burst detection algorithm does not catch all bursts.
- **Burst Period as Feature (BPF)**: Tweets in burst periods are labeled as important and therefore has increased weight. We do this by adding weight to original importance of the tweets in the burst periods. This method ensures no loss of information by keeping all the tweets as an input.

For both methods, each tweet in a tweet stream has importance calculated through collection frequency (CF)[1] weighting. We first calculate the importance of each term consisting of a tweet based on CF weighting, and then add up all importance of terms consisting of the tweet.

For selecting important tweets from among all tweets in the stream, we use a maximum coverage problem with knapsack constraint model (MCKP) [4]. This model is to cover terms in input tweets as many as the given maximum length of an output summary allows. Due to the characteristics that this model does not take into account the importance of words more than once, it is invulnerable for redundant inputs; hence this model has widely been used in the area of multi-document summarization where severe redundancy prevails. Due to a huge amount of input tweets, we employ an approximation method, a greedy algorithm, to find a set of important tweets. We use Shuca[2] as an implementation of MCKP to employ it.

Input	
Average number of tweets	37,137
Average tweets/min	275
Reference summaries	
Average word count	464
Average sentence count	15

Table 1: Corpus statistics

4 Evaluation

4.1 Data

In order to evaluate our system we use a data set consisting of 16 soccer matches of Barclays Premier League between August and December 2015. We use Twitter's stream API to record tweets. Updates for each match are filtered from Twitter stream by using a league and team name as a keyword. We also prepare reference summaries taken from premierleague.com[3] to use as gold standard. The dataset's statistics are shown in Table 1.

4.2 Preprocessing

After extracting texts from tweet data, we remove all non-word characters, emoticons, and URLs. We then use Porter's algorithm[4] in the stemming process.

4.3 Parameter Settings

In our burst detection program, we calculate tweet volume for each match, then set the burst threshold to three times the median value. Time window for tweet volume calculation is set to one minute. We also set the length of generated summary to 15 sentences which is the same as the average length of reference summaries.

4.4 Evaluation Method

We use ROUGE-1 [6] as an evaluation method. Not only input tweets but also reference summaries are

[1]If term A appears 1,000 times in an input tweet stream, its CF is 1,000. This scoring method emphasis on terms appearing frequently.

[2]`https://github.com/hitoshin/shuca`

[3]`http://www.premierleague.com/`

[4]`http://tartarus.org/martin/PorterStemmer/`

	Random	CF	BPS	BPF
ROUGE-1	0.106	0.275	0.248	0.269

Table 2: Average ROUGE score

	Recall	Precision
Average	0.377	1

Table 3: Burst Detection Evaluation

stemmed for matching terms among output summaries with those among reference summaries.

4.5 Compared Method

We compared following four methods:

- **Random**: Tweets for an output summary are randomly selected. In this method, a summarizer try to select a tweet randomly from among an input tweet stream until the length of selected tweets reaches a given maximum summary length. This is the first baseline.
- **CF**: Tweets are selected through MCKP with CF weighting. That is, this method does not employ burst detection. This is the second baseline.
- **Burst Period as Segment (BPS)**: Tweets are extracted from only burst periods and then used to generate a summary. This is a proposed method.
- **Burst Period as Feature (BPF)**: Tweets in burst periods obtain additional weight. This is also a proposed method.

4.6 Summarization Results

Average ROUGE scores for matches are shown in Table 2. CF performed best, then BPF and BPS followed. We discuss this result in the next section.

4.7 Discussion

From the experiment result, we notice a drop in ROUGE score when burst detection is applied to the summarization process, especially when we use the BPS method. This is most likely because our burst detection program is not performing well enough.

In order to evaluate the burst detection process, we manually annotated bursts for each match based on reviewing tweet volume changes as well as match reports. Each burst period is represented as a pair of time points $\{start\ time, end\ time\}$. Then we compare this with the result of our burst detection program. The average recall and precision score are

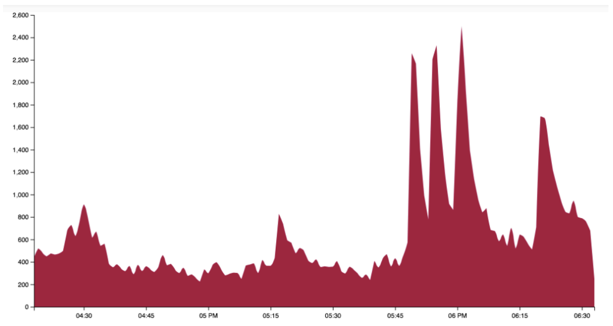

Figure 1: Match showing clear spikes

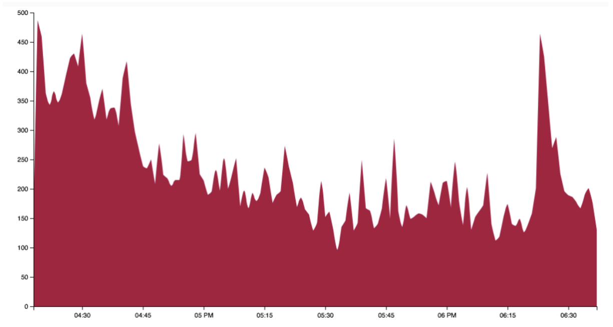

Figure 2: Match showing many small spikes

shown in Table 3. Recall and precision score of our burst detection algorithm show that although it successfully identified bursts in the stream, it failed to catch all burst on the stream. Since the proposed BPS method only generates a summary from the tweets in burst periods, the result is more affected by burst detection performance, showing the lower score.

We show an example of bursts in Figure 1 and 2. In Figure 1, clear three spikes are observed in the latter of the match. In such a case our burst detection algorithm easily identify bursts among the stream. In Figure 2, however, many small spikes are observed, inherently making burst detection erroneous.

In order to evaluate the effectiveness of applying burst detection to summarization, we redo the experiment with a part of the data set using human-annotated bursts instead of our burst detection program. The results are shown in Table 4. The number of bursts shown in Table 4 is manually counted. We notice that even though the gold standard for burst detection is used, there are cases which the score does not improve comparing to our baseline method without burst detection.

In those cases, our hypothesis is that what Twitter users view as important information does not always coincides with journalists. Twitter users seem to be more interested in what is currently happening and are more likely to react to events such as goal or

Match	Random	CF	BPS	Bursts
Ars-Sto	0.075	0.284	0.308	3
Che-Sto	0.104	0.265	0.258	1
Eve-Liv	0.071	0.304	0.263	3
Lei-Mun	0.166	0.262	0.270	3
Sun-Tot	0.191	0.331	0.304	2
Swa-Eve	0.057	0.274	0.281	2
Tot-Che	0.132	0.251	0.305	3
Wat-Ars-	0.112	0.260	0.292	4
Wat-Cry	0.162	0.324	0.333	4
Wat-Mun	0.136	0.243	0.269	4

Table 4: ROUGE score with human-annotated bursts

red card in a soccer match. This is proven by how bursts often coincide with those events. However, by examining news reports we used as gold standard in the experiment, news writers on the other hand are also interested in other details and they often view the match as a whole. We also notice that matches in which the score failed to improves are ones with less sub-events, such as goals, and fewer bursts are detected, less than 4, in our case.

5 Conclusion

In this research, we attempt to apply burst detection in order to improve microblog summary generation. By examining the result, we conclude that although there are improvements in many cases as long as the burst detection algorithm performs well enough, applying burst detection does not always guarantee a better output summary. However, since using burst detection to extract tweets for summarization can drastically reduce input size, it has much better time efficiency while being able to provide comparable results in many cases.

As for future work, based on the observation described above, separating the streams where burst detection would work from those where it would not work is expected to be a promising direction.

References

[1] D. Chakrabarti and K. Punera, Event Summarization Using Tweets, In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*, pp. 66–73, 2011.

[2] J. Kleinberg, Bursty and Hierarchical Structure in Streams, In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 91–101, 2002.

[3] M. Kubo, R. Sasano, H. Takamura, M. Okumura, Generating Live Sports Updates from Twitter by Finding Good Reporters, In *Proceedings of the 2013 IEEE/WIC/ACM International Conferences on Web Intelligence (WI) and Intelligent Agent Technology (IAT)*, pp. 527–534, 2013.

[4] H. Takamura and M. Okumura, Text Summarization Model based on Maximum Coverage Problem and its Variant, In *Proceedings of the 12th Conference of the European Chapter of the ACL*, pp. 781–789, 2009.

[5] J. Nichols, J. Mahmud, and C. Drews, Summarizing sporting events using twitter, In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces (IUI)*, pp. 189–198, 2012.

[6] C. Y. Lin, ROUGE: A Package for Automatic Evaluation of Summaries, In *Proceedings of Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pp. 74–81, 2004.

[7] B. Sharifi, M. A. Hutton, and J. Kalita, Summarizing microblogs automatically, In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL (HLT-NAACL)*, pp. 685–688, 2010.

[8] H. Takamura, H. Yokono, and M. Okumura, *Summarizing a document stream*, In *Proceedings of the annual 33rd European Conference in Information Retrieval (ECIR)*, pp. 177–188, 2011.