Realtime Ad Hoc Search in Twitter: Know-Center at TREC Microblog Track 2011

Christopher Horn, Oliver Pimas, Michael Granitzer, and Elisabeth Lex

Know-Center Graz, Inffeldgasse 21a, 8010 Graz, Austria
{chorn,opimas,mgrani,elex}@know-center.at

Abstract. In this paper, we outline our experiments carried out at the TREC Microblog Track 2011. Our system is based on a plain text index extracted from Tweets crawled from twitter.com. This index has been used to retrieve candidate Tweets for the given topics. The resulting Tweets were post-processed and then analyzed using three different approaches: (i) a burst detection approach, (ii) a hashtag analysis, and (iii) a Retweet analysis. Our experiments consisted of four runs: Firstly, a combination of the Lucene ranking with the burst detection, and secondly, a combination of the Lucene ranking, the burst detection, and the hashtag analysis. Thirdly, a combination of the Lucene ranking, the burst detection, the hashtag analysis, and the Retweet analysis, and fourthly, again a combination of the Lucene ranking with the burst detection but in this case with more sophisticated query language and post-processing. We achieved the best MAP values overall in the fourth run.

1 Introduction

Over the past years, the microblogging platform Twitter has gained momentum since people use this platform to share information about current events and about their daily life. In June 2011, Twitter released the information that now 200 million Tweets are sent over Twitter per day[1]. Consequently, it is challenging to find relevant Tweets in this huge amount of data.

Typically, finding relevant Tweets is considered as ad hoc search where users formulate their information needs by means of a query. However, this does not address temporal aspects - for instance, how novel is a Tweet or the information provided in a Tweet.

In the TREC Microblog Track 2011, candidate Tweets have to be ranked not only by topical relevance to a query but also by temporal aspects. More specifically, most recent relevant Tweets should be ranked higher[2].

In our approach towards the TREC Microblog Track 2011 we combine a topical relevance ranking with three different scores: (i) a Burst Detection Score to include temporal aspects of Tweets, (ii) a Hashtag Score to improve the relevance ranking, and (iii) a Retweet Score to identify influential Tweets.

[1] http://blog.twitter.com/2011/06/200-million-Tweets-per-day.html
[2] https://sites.google.com/site/microblogtrack/2011-guidelines

2 Task

The goal of the TREC Microblog Track 2011 is to perform realtime ad hoc search in Twitter. Firstly, an ad hoc search task has to be carried out where user information needs are expressed by a set of queries at a specific timestamp. This search task results in a list of Tweets that are thematically relevant to a query. The real time aspect is addressed so that not only relevant but the most recent Tweets to a query are to be ranked highest. As a consequence, to each query, a list of relevant Tweets should be returned that is ordered from most recent to oldest.

3 Collection

For the TREC Microblog Track 2011, the Tweets2011 corpus is used that contains 16 million Tweets from a time period of two weeks (24th January 2011 until 8th February, inclusive). The Tweets2011 corpus contains a variety of Tweets: high quality and spam Tweets, replies and Retweets. Therefore, it reflects a real sample of the Twittersphere.

In the corpus, each day is represented by a block of about 10,000 chronologically ordered Tweets compressed using gzip. The Tweets are available in both JSON format as well as in HTML format.

We directly downloaded the corpus from Twitter using the HTML crawler from the twitter-corpus-tools[3] provided by the challenge organizers. Consequently, our Tweets are in HTML format.

4 Approach

We first indexed the 16 million Tweets using the twitter-corpus-tools. This resulted in a search index of size 4.1 GB. Then, we searched the index for the 50 topics that have been provided by the challenge organizers. All topics are assigned a timestamp; therefore each query represents a specific information need at a specific time. As indexing and search framework, we used Apache Lucene[4], an open source search engine that is able to index vast amounts of documents at reasonable time. Lucene enables to carry out complex searches and Boolean queries very efficiently; also the search results are already ranked by their relevance to the search query.

Many of the search topics consist of more than one term; the default setting in Lucene is that the single terms of such a phrase are combined with a Boolean OR. However, this often is not precise enough to retrieve correct relevant Tweets. Therefore, we combined the terms following several search strategies that are described in the next section.

[3] `https://github.com/lintool/twitter-corpus-tools`
[4] `http://lucene.apache.org`

4.1 Search Strategies

If the query topic consisted of more than one term and it contained a year, e.g. 2022, we extracted the year with regular expressions and combined it and the rest of the query terms with a Boolean AND since a year typically denotes an important event. For example, if the topic is 2022 FIFA soccer, the according Lucene search query becomes:

(2022 AND (FIFA OR soccer)) AND *Timestamp*.

Note that *Timestamp* denotes the *"timestamp of the query in terms of the chronologically nearest Tweet id within the corpus"*[5].

If the topic consists of more than two query terms, we define that least two query terms must occur. For example, if the topic is NIST computer security, the according Lucene search query becomes:

(NIST AND (computer OR security)) AND Timestamp

However, if with this search query no or only a few results were retrieved, we repeated the search combining the query terms using OR:

(NIST OR (computer OR security)) AND Timestamp

Finally, the relevant Tweets resulting from particular search queries then were subject to several post-processing steps. These are described in the next section.

4.2 Tweet Post-Processing

Since the collection contains not only English Tweets, we implemented a custom language guesser to filter out Tweets in other languages. As stated by the challenge organizers, only English Tweets are relevant. Standard language guessers did not work on the Tweet corpus due to the nature of Tweets: they are only 140 characters long and typically feature a very domain specific language like e.g. abbreviations.

Our custom language guesser first removes the links from the Tweets since most links contain English terms that would consequently confuse the language guesser. The language guesser converts the remaining Tweet text to bytes and for each byte, it checks whether the character is an ASCII character or not. Finally, the ratio between the number of ASCII characters and total characters is computed. If this ratio is below 0.5 - which means that more than the half of characters are non ASCII characters - the Tweet is removed from the search results. This enables us to filter out Chinese, Arabic etc. Tweets. Additionally, we use Spanish, French, and German stop words to filter Tweets in these languages.

[5] `https://sites.google.com/site/microblogtrack/2011-guidelines`

Aside from the language, we sort out Tweets that contained emoticons, double exclamation marks, and double question marks since these character combinations are clearly an indicator for the Tweet being opinionated [5] and therefore of lower quality [6].

Also, we remove all Tweets from the search results that contain the character @. Such Tweets are typically replies to other Twitter users. As stated by the challenge organizers, replies are considered as being not relevant to a query. To filter out Tweets that simply repeat information originally posted by others and not marked as Retweet, we compute the Levenshtein distance between the current Tweet and the last Tweet whereas they must differ by at least 50% of their characters.

After the post-processing step of the search results, we calculate three additional ranking scores, namely (i) the Burst Detection Score, (ii) the Retweet Score, and (iii) the Hashtag Score to retrieve not only relevant, but also the most interesting Tweets. These three scores are described in the next section.

4.3 Burst Detection Score

Content published over Twitter is a as continuous stream of Tweets on arbitrary topics or comments in real time. Due to Twitter's highly dynamic nature, many topics arise, grow in intensity for a certain amount of time and eventually fade away. The identification of a new topic in a stream of Tweets is useful to determine the importance of a Tweet. We assume that a Tweet is more important if it is the first to discuss a highly influential topic like the Egyptian Revolution opposite to Tweets that simply Retweet and redistribute the same information. Finding influential Tweets can be addressed with burst detection. When determining the appearance of a topic in a stream of documents, this is referred to as *"burst of activity"* [3].

In our approach, we implement the burst detection by dividing the covered time range of a certain topic into different burst windows. Then, we count how many Tweets each window contains. If (i) the count exceeds a given threshold, and if (ii) compared to the previous window, the count increases by a certain amount, the particular window is considered to be a burst. All Tweets within that burst window are regarded to be more relevant than Tweets outside this window, hence we rank them higher. Also, we consider the case where a burst falls in between two burst windows. Therefore, we overlap the burst windows by a certain amount of time.

Table 1 shows the parameters we used for our burst detection approach.

MB009, Title: Toyota Recall

Toyota announced to recall 1.7 million cars due to various issues. Reuters reported about the recall on January 26 2011 at 12:30 CET[6]. Our system detected a burst at the same time:

[6] http://www.reuters.com/article/2011/01/26/us-toyota-recall-idUSTRE70P2EC20110126

Parameter	Value	Description
Window size	3 hours	Size of the burst windows (in hours)
Overlapping time	1 hour	Amount in hours by which windows are overlapping
Burst threshold	4	Necessary amount of Tweets within that window for a window to be considered as burst.
Minimum increase	2	Necessary increase compared to the previous window

Table 1. Parameters used for Burst Detection

```
Burst with 26 Tweets detected between
   Wed Jan 26 08:13:49 CET 2011 and Wed Jan 26 12:13:49 CET 2011
Burst with 25 Tweets detected between
   Wed Jan 26 17:13:49 CET 2011 and Wed Jan 26 21:13:49 CET 2011
```

MB001, Title: BBC World Service staff cuts

On January 26, 2011, BBC announced to cut 650 jobs in the World Service department[7]. Our system detected a burst with 19 Tweets on January 26 2011 regarding this topic:

```
Burst with 19 Tweets detected between
  Wed Jan 26 09:01 CET 2011 and Wed Jan 26 13:01 CET 2011
Burst with 12 Tweets detected between
  Wed Jan 26 18:01 CET 2011 and Wed Jan 26 22:01 CET 2011
```

Topic MB020, Query: Taco Bell filing lawsuit

On January 24, 2011, a lawsuit against Taco Bell was filed, asking to explain how much actual meat is contained in their meat. Amongst other media, the Chicago Tribune reported this issue on Jan 25 2011, 01:12 CET[8], which corresponds to the output of our burst detections:

```
Burst with 9 Tweets detected between
  Mon Jan 24 17:20:35 CET 2011 and Mon Jan 24 21:20:35 CET 2011
Burst with 13 Tweets detected between
  Tue Jan 25 05:20:35 CET 2011 and Tue Jan 25 09:20:35 CET 2011
Burst with 43 Tweets detected between
  Tue Jan 25 20:20:35 CET 2011 and Wed Jan 26 00:20:35 CET 2011
```

[7] http://www.bbc.co.uk/news/entertainment-arts-12283356

[8] http://www.chicagotribune.com/news/opinion/ct-talk-taco-bell-0125-20110124,0,4971444.story

Topic MB021, Title: Emanuel residency court rulings

At this given time, the breaking news was spread that Emanuel Rahm is not allowed to candidate as major for Chicago[9]. Our system detected a large amount of 48 Tweets within a burst window.

```
Burst with 48 Tweets detected between
  Mon Jan 24 18:44:38 CET 2011 and Mon Jan 24 22:44:38 CET 2011
```

These examples reveal that our burst detection approach is feasible to find trending topics in a given data set.

4.4 Hashtag Score

Hashtags serve as indicators for a Tweet's meaning, intended audience, or topic [2]. In the past, hashtags have even be used to initiate specific events as for example the recent protests against Wallstreet[10].

In this work, we derived the most prominent hashtag for a topic. We assume that a Tweet is more relevant if it contains the most important hashtag. We assigned every Tweet that contained the most prominent hashtag a score of 1 and 0 otherwise. To identify this hashtag, we created a hashtag stop word list consisting of hashtags used by advertisments, namely #ad, #sales, hashtags used to gather followers, namely #ff, #followme, and hashtags that are automatically created by music player applications, namely #nowlistening, #nowplaying, #np, #lastfm, #itunes. Also, we included the hashtag #wtf since it is often used and it does not carry topic-relevant information.

4.5 Retweet Score

In Twitter, Retweets are used to repeat a piece of information. Therefore, the amount of Retweets can be exploited to judge the importance and influence of both a Tweet and a Twitter user [1]. Since we used the HTML version of the Tweets, the Retweet count per Tweet was not directly available; therefore, we had to calculate it on our own. To calculate the number of Retweets per Tweet, we used the textual structure of Retweets on twitter.

For a certain Tweet t, we searched the whole corpus for the matching expression $e =$ "@ RTauthor-name" where *author-name* denotes the name of the author of t. This resulted in a collection of all Retweets R from that author within our corpus. In order to reduce this collection to only the Retweets of t, the trailing text after the matched expression e must match the original Tweet t. Since Tweets are limited to 140 characters and the length of Retweets is further limited due to Twitter's Retweet convention, i.e. expression e, a Retweet r may only repeat the first n characters of the original Tweet t.

[9] http://www.suntimes.com/3469419-417/ballot-booted-court-emanuel-rahm.html

[10] http://tribune.com.pk/story/276605/occupy-wall-street-from-a-single-hashtag-a-protest-circled-the-world/

So we defined r as being a Retweet of t based on two conditions: Firstly, r must contain e. Secondly, all trailing characters of e in r have to match the first n characters of t, n being the number of characters following the expression e in r. In other words, t is not required to be equal to but to contain the trailing text of e in r.

This method results in a list of Tweets and a counter how often they were retweeted. To finally get a score from our Retweets analysis, we assigned every Tweet a score between 0 and 1 (i.e. the Retweet Score), relatively to the number how often it was retweeted.

In our work, we used the whole corpus to calculate the Retweets, which consequently represents future evidence. However, for a real-time task, only Retweets that existed at the time the query was submitted really should be used.

4.6 Runs

We submitted four different runs: (i) Run 1, (ii) Run 2, (iii) Run 3, and (iv) Run 4. These runs are described in detail in the next sections.

Run 1 For Run 1, we first used Lucene to query and rank the Tweets. After that, we carried out the burst detection, identifying the time windows (interval: 3 hours) when an unusual amount of Tweets occurred. Based on our assumption that something extraordinary happened, those Tweets have been ranked higher by the Burst Detection Score given in Equation 1:

$$FinalBurstScore = LuceneScore * 0.51 + BurstDetectionScore * 0.49; \quad (1)$$

where the $BurstDetectionScore$ is 1 if a Tweet is in the burst window and 0 otherwise. Our final score combines the Lucene Score with the Burst Detection Score where we assign the Lucene Score a higher priority. If no burst was detected, only the Lucene score has been used to rank the Tweets.

Run 2 For Run 2, we again first used Lucene to query and rank the Tweets. After that, we performed the burst detection, identifying the time windows (interval: 3 hours) to compute the Final Burst Score. As second ranking factor, we counted the number of Retweets (Retweet Score) of all Twitter user over the whole corpus. Our final score combines therefore the Final Burst Score and the Retweet Score.

$$Score = FinalBurstScore * 0.8 + RetweetScore * 0.2; \quad (2)$$

Run 3 For Run 3, we again first used Lucene to query and rank the Tweets by topical relevance. After that, we performed the burst detection, identifying the time windows (interval: 3 hours) when an unusual amount of Tweets occurred to compute the Final Burst Score. As second ranking factor, we computed the

most often used hashtag per topic and ranked Tweets higher that contained the most often used hashtag (Hashtag Score). Note that this computation has been done only on Tweets posted before the query timestamp (no future evidence). Our final Score combines therefore the Final Burst Score, and the Hashtag score:

$$Score = FinalBurstScore + HashtagScore * 0.005; \quad (3)$$

Run 4 In Run 4, we again first used Lucene to query and rank the Tweets. After that, we performed the burst detection to compute the Final Burst Score. The final Score is computed as given in Equation 1. The difference to Run 1 is that we used the Levenshtein distance [4] computed between subsequently ranked Tweets to remove Retweets that do not follow the Twitter Retweet convention in the final ranking. Note that subsequently ranked Tweets must differ by at least 50% of their characters. Also, in this run, we used more sophisticated stop word lists considering slang words, emoticons, hashtags with no semantic meaning, etc.

5 Results

The overall MAP results achieved for all four runs are presented in Table 2: The best results in terms of MAP over all topics were achieved in Run 4. So

Run	MAP
1	0.1781
2	0.1905
3	0.1905
4	0.1919

Table 2. MAP results for all four runs.

apparently, exploiting only the Lucene relevance ranking in combination with our burst detection approach gives the best results. The second best results were achieved in Run 2 and Run 3. In the following, we concentrate on Run 3 since in Run 2, future evidence has been used. More specifically, we compare Run 3 with 4 in respect to a subset of the given topics. Table 3 details results achieved in Run 3 and Run 4 for a selection of topics:

Our approach performed very well for some queries: for instance, for topic MB007, we achieved a precision on the 30 highest ranked topics (P@30) of 86% in Run 3. The median P@30 over all submissions is 60%. As can be derived from the table, including the hashtag score did not improve the results for this topic. This can be interpreted that the relevant Tweets did not contain the most prominent hashtag or that no hashtag existed for the relevant Tweets.

In case of topic MB003, no burst was detected but the hashtag analysis improved the P@30 from 0.6667 (Run 4) to 0.7000 (Run 3.). The hashtag in this case was #haiti, a clearly topically related term.

Topic	Query	P30 Score Run 4	P30 Score Run 3	Burst found
MB007	Pakistan diplomat arrest murder	0.8667	0.8667	yes
MB008	phone hacking British politicians	0.5333	0.5333	yes
MB036	Moscow airport bombing	0.6000	0.6000	yes
MB041	Obama birth certificate	0.3333	0.3333	yes
MB003	Haiti Aristide return	0.6667	0.7000	no

Table 3. Results for Run 3 and Run 4 for selected topics (results taken from allrel).

We investigated the resulting MAP values for our four runs. For topic MB021, we achieved a high MAP value of 0.2794. For this topic, we detected 7 bursts. In this case, our burst detection approach was therefore successful.

For topic MB048, we achieved a low MAP of 0.0040. We investigated the results of our approach and we found that for this topic, a large amount of bursts (18 bursts) were detected by our system. We checked the Tweets contained in the bursts and we found that the Tweets tackled the Egyptian revolution but not the specific topic Egyptian evacuation. In this case, with the burst detection, we identified the general topic of the Egyptian revolution, but not the subtopic of the evacuations. Therefore, our approach was not successful in that specific case. The challenge here is how to identify the most relevant bursts.

6 Conclusions

In conclusion, we presented our approach towards the TREC Microblog Track 2011. We proposed a retrieval technique in combination with three approaches: a burst detection, a Retweet analysis, and a hashtag analysis. Our experiments revealed that for certain topics, we achieve high MAP values using the burst detection. If too many bursts are detected, it is challenging to identify which bursts are more relevant than others. In our setting, we treated all bursts equally. For future work, we aim to distinguish relevant bursts from non relevant bursts. Also, for certain topics, incorporating the hashtag score was beneficial: frequently used hashtags indicate the semantic relation to a topic. However, boosting Tweets with the wrong hashtags negatively influences the ranking. So, the identification of hashtags relevant for topics is crucial.

7 Acknowledgements

The Know-Center GmbH Graz is funded within the Austrian COMET Program - Competence Centers for Excellent Technologies - under the auspices of the Austrian Ministry of Transport, Innovation and Technology, the Austrian Ministry of Economics and Labor and by the State of Styria. COMET is managed by the Austrian Research Promotion Agency FFG.

References

1. D. Boyd, S. Golder, and G. Lotan. Tweet, tweet, retweet: Conversational aspects of retweeting on twitter. *Hawaii International Conference on System Sciences*, 0:1–10, 2010.
2. M. Efron. Hashtag retrieval in a microblogging environment. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 787–788, New York, NY, USA, 2010. ACM.
3. J. Kleinberg. Bursty and hierarchical structure in streams. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 91–101, New York, NY, USA, 2002. ACM.
4. V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.
5. E. Lex, M. Granitzer, M. Muhr, and A. Juffinger. Stylometric features for emotion level classification in news related blogs. In *Proceedings of the 9th RIAO Conference (RIAO 2010)*, 2010.
6. E. Lex, A. Juffinger, and M. Granitzer. Objectivity classification in online media. In *HT '10: Proceedings of the 21st ACM conference on Hypertext and hypermedia*, pages 293–294, New York, NY, USA, 2010. ACM.