IRIT at TREC Microblog 2012: Adhoc Task

Lamjed Ben Jabeur, Firas Damak,
Lynda Tamine, Karen Pinel-Sauvagnat, Guillaume Cabanac, and Mohand Boughanem

{jabeur, damak, tamine, cabanac, sauvagnat, boughanem}@irit.fr,
IRIT/SIG
118 route de Narbonne F- 31062 Toulouse cedex 9

Abstract. This paper describes the participation of the IRIT lab, university of Toulouse, France, to the Microblog Track of TREC 2012. Two different models are experimented by our team for the adhoc task: (i) a Bayesian network retrieval model for tweet search and (ii) a feature learning model for relevance classification. Experimental results show that Bayesian network retrieval model improves the performances comparing to the track median.

1 Introduction

Microblogs are popular networking services that enable users to broadcast information. They emerge as a promising tool to get acquainted with the latest news. However, seeking for information over microblogs becomes a challenging task due the increasing amount of published information. In the case of Twitter[1] microblogging service, about 340 million[2] tweets are published every day. Part of these tweets are useless, ambiguous, redundant or incredible [1]. A new information retrieval task is therefore created. Its main purpose is to search for real-time information and to rank recent tweets. TREC 2011 Microblog track [2] defines tweet search as a real-time adhoc task where the users are interested in most recent and relevant information. In spite of Web search, tweet search aims to find temporally relevant information, monitor content and follow current events and people activities [3].

Prior works addressing tweet search integrate a variety of textual features, microblogging features, spatiotemporal features and social network features [4, 5]. These works consider that tweet relevance depends, on the one hand, on its publishing context, and on the other hand, on the content quality such as URLs, mentions and hashtags. We investigate in this paper two different approaches for microblog retrieval:

- First, a bayesian network retrieval model for tweet search estimates the tweet relevance based on the microblogger influence and the time magnitude. The influence score is computed by applying PageRank algorithm on the social network of retweets and mentions. The time magnitude is estimated from the set of tweets in the same period that contains similar query terms.
- Second, a machine learning for microblog retrieval integrates a variety of features. TREC 2011 topic results were used as a learning set.

Both approaches concern the real-time adhok task. This paper is organized as follows. Section 2 introduces the Bayesian network model for tweet search and discusses associated results. Section 3 describes the learning model for microblog retrieval and compares different strategies for tweet classification.

[1] http://www.twitter.com/
[2] http://blog.twitter.com/2012/03/twitter-turns-six.html

2 A Bayesian network retrieval model for tweet search

Tweet search is a particular information retrieval task driven by a variety of topical, social and temporal motivations [3]. Inspired from work of De Cristo et al. [6] that proposes to integrate topical and hyperlink-based authority evidences into a Bayesian belief network, we propose to model tweet search using Bayesian network models that incorporate different sources of evidence into an integrated framework.

2.1 Bayesian network topology

We describe in figure 1 the topology of our Bayesian network model for tweet search. The Bayesian network model for tweet search is comprised of 3 connected networks:

- *Tweet network:* Each term k_i is represented as a node in the term layer K. A user query is modeled by a node q. A tweet t_j is represented by three nodes t_{kj}, t_{sj} and t_{oj} which belong to the topical evidence layer TK, the social evidence layer SO and the temporal evidence layer TS, respectively. These nodes are connected to a another node t_j from tweet layer T. We notice that q and t_{kj} are the only nodes connected to terms layer K.
- *Microblogger network:* Each microblogger u_f is represented by a node in the social layer S. Microbloggers nodes are connected to relative tweet nodes in the social evidence layer TS and corresponding term nodes in layer K.
- *Period network:* Each period o_e is represented by a node in the temporal layer O. Periods are connected to respective tweets from temporal layer TO and corresponding terms in layer K. A period is defined with a date θ_{o_e} and a time window Δt.

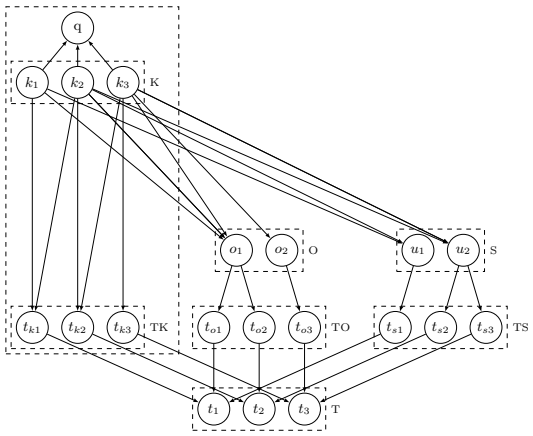

Fig. 1. Belief network model for tweet search [7]

2.2 Query evaluation

The relevance of a tweet t_j with respect to a query q submitted at θ_q is computed by the probability $P(t_j|q, \theta_q)$. Based on the topology of the Bayesian network for tweet search and ignoring the query date, the probability $P(t_j|q)$ is developed as follows:

$$P(t_j|q) \propto \sum_{\boldsymbol{k}} P(q|\boldsymbol{k})P(t_{kj}|\boldsymbol{k})P(t_{sj}|\boldsymbol{k})P(t_{oj}|\boldsymbol{k})P(\boldsymbol{k}) \tag{1}$$

$\boldsymbol{k}$ is a term configuration. To simplify the computation of probability $P(t_j|q)$, only instantiated terms in the query are considered in the configuration $\boldsymbol{k}$. Relevance probability tweets with corresponding date θ_{t_j} is posterior to query date θ_q is set to $P(t_j|q) = 0$. $P(\boldsymbol{k}) = \frac{1}{2^n}$. n is the number of query terms.

Let $w_{k_i} = \frac{df_{k_i}}{N}$ be the weight of term in the collection. df_{k_i} is the number of tweets containing k_i and N is the number of posterior tweets to the query q. The probability $P(q|\boldsymbol{k})$ highlights configurations with significant rare terms as follows:

$$P(q|\boldsymbol{k}) = \begin{cases} \frac{1 - \prod\limits_{k_i \in c(\boldsymbol{k}) \wedge q} w_{k_i}}{1 - \prod\limits_{k_i \in q} w_{k_i}}, & \text{if } c(\boldsymbol{k}) \neq \emptyset \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

let $w_{k_i,t_j} = \frac{tf_{k_i,t_j} - \beta}{tf_{k_i,t_j}}$ be the term wight of k_i in t_j that map high frequencies into a small interval. We note that small value of β reduces the weight of frequent terms. Accordingly, we give less importance to term frequency rather than term presence in the case of long queries. The probability $P(t_j|\boldsymbol{k})$ is finally computed as:

$$P(t_j|\boldsymbol{k}) = \begin{cases} \frac{\sum\limits_{k_i \in c(t_j) \wedge c(\boldsymbol{k})} w_{k_i,t_j}}{\sum\limits_{\boldsymbol{k}} \sum\limits_{k_i \in c(t_j) \wedge c(\boldsymbol{k})} w_{k_i,t_j}}, & \text{if } c(t_j) \wedge c(\boldsymbol{k}) \neq \emptyset \\ \delta, & \text{otherwise} \end{cases} \tag{3}$$

δ is a default probability.

Assuming that the two events of observing microblogger u_f and configuration $\boldsymbol{k}$ are independent, we write:

$$P(t_{sj}|\boldsymbol{k}) = P(t_{sj}|u_f)P(u_f) \tag{4}$$

Let $\tau(u_f)$ be the set of tweets published by u_f. $P(t_{sj}|u_f) = \frac{1}{|\tau(u_f)|}$. Probability $P(u_f)$ is approximated to $PageRank(u_f)$, the microblogger PageRank score computed on the social network of retweets and mentions extracted from the instantiated tweets by the query.

The probability $P(t_{oj}|\bar{o}_e)$ of observing the tweet outside the respective period is equal to 0. Thus, $P(t_{oj}|\boldsymbol{k})$ is written as:

$$P(t_{oj}|\boldsymbol{k}) = P(t_{oj}|o_e)P(o_e|\boldsymbol{k}) \tag{5}$$

Let $\tau(o_e)$ and $\rho_{o_e}(t_j)$ be the set of corresponding tweets and retweets of t_j in period o_e. The first probability is computed as follows: $P(t_{oj}|o_e) = \frac{1+|\rho_{o_e}(t_j)|}{|\tau(o_e)|}$. To highlight active period of the configuration $\boldsymbol{k}$ that concurs with a real world event, periods are weighted as following:

$$w_{o_e,\boldsymbol{k}} = \frac{\log(\theta_q - \theta_{o_e})}{\log(\theta_q - \theta_{o_s})} \times \frac{df_{\boldsymbol{k},o_e}}{df_{\boldsymbol{k}}} \tag{6}$$

θ_q, θ_{o_e} and θ_{o_s} are respectively the timestamps of the query q, the period o_e and the period o_s when the oldest tweet containing the term configuration $\boldsymbol{k}$ is published with $\theta_{o_s} \leq \theta_{o_e} \leq \theta_q$. $df_{\boldsymbol{k},o_e}$ is the number of tweets published in o_e and containing the configuration $\boldsymbol{k}$. $df_{\boldsymbol{k}}$ is the number of tweets with the term configuration $\boldsymbol{k}$.

The probability $P(o_e|\boldsymbol{k})$ is computed as:

$$P(o_e|\boldsymbol{k}) = \begin{cases} \frac{w_{o_e,\boldsymbol{k}}}{\sum\limits_{\boldsymbol{k}} w_{o_e,\boldsymbol{k}}}, & \text{if } df_{\boldsymbol{k},o_e} > 0 \\ \delta, & \text{otherwise} \end{cases} \tag{7}$$

2.3 Results and discussion

Table 1 compares results presented by different configurations of our model. *IRITbnetKSO* represents our Bayesian network model with all features activated. *IRITbnetK* represents our model with only the topical feature is activated. *IRITbnetKS* and *IRITbnetKO* are based on the topical feature and represents our model with the social feature is activated and the temporal feature is activated, respectively. First, we note that all configuration results overpass the TREC median. The social feature presented by *IRITbnetKS* configuration does not improve the performances of topical baseline *IRITbnetK*. This is explained by the low density of extracted social network with few retweet associations. This problem has also affected the results of *IRITbnetKSO* configuration. In contrast, the temporal feature improves the topical relevance for main measures (p@30, map) which highlight the importance of temporal context in microblog retrieval. Considering the ROC curves, we notice that similar tendencies are shown by all configurations.

	p@10	p@20	p@30	p@100	MAP	R-Precision
IRITbnetK*	**0.2610**	**0.2110**	0.1983	0.1363	0.1715	**0.2035**
IRITbnetKS	0.2407	0.2102	0.1831	0.1337	0.1681	**0.2035**
IRITbnetKO	0.2305	0.2085	**0.1994**	**0.1395**	**0.1742**	0.2019
IRITbnetKSO*	0.2322	0.2076	0.1960	0.1386	0.1717	0.2025
TREC median			0.1808		0.1486	0.1869

Table 1. Comparison of model configurations. * Official run

3 Learning features for microblog search

We describe in this section an approach that learn a set features for relevance classification. We used the 2011 track topic results as a learning set. We also crawled and indexed the titles of web pages published in the tweets to compensate their shortness.

3.1 Design of our approach

Our approach follows several steps:

- We first index the collection and retrieve the top-1500 relevant tweets for each topic using a search engine as described in section 3.1.1.
- We then process the outcome tweets to calculate some feature scores (Section 3.1.2).
- The next step consists in classifying resulting tweets using a learning model. Only those that have been classified as relevant are kept (Section 3.1.3).
- Finally, before displaying results, we processed the resulting tweets with a language filter so that only those tweets written in English would be delivered to the user[3].

[3] http://code.google.com/p/language-detection/

3.1.1 Indexing and retrieving

We chose to use the Lucene platform[4] in our approach. We specified multiple fields to index the corpus: ID of the tweet, AUTHOR, HTTPSTATUSCODE, TEXT, and URLTITLE. The aim of storing the HTTPSTATUSCODE is to be able to retrieve only original tweets (HTTPSTATUSCODE equals to 200). We dropped other tweets (HTTPSTATUSCODE equals to 302, 403, and 404) since it was announced that retweets would not be judged as relevant. The content of tweets was indexed in the TEXT field. The field URLTITLE was used to index titles of web pages published in tweets. We made some modifications in Lucene search engine: first, we integrated the BM25 model in addition to VSM that exists by default. Second, we modified the scoring functions to be able to consider only tweets published before the query time in the index, when calculating scores and when retrieving tweets.

In the rest of the paper, the corpus of $top - 1500$ relevant tweets obtained by the Lucene search engine regarding a topic q is denoted by Tq, Cq denotes the corpus of all tweets published before timestamp of a topic q. ($Tq \subseteq Cq$). Finally, apart from indexing and retrieving, we used the Lucene search engine to calculate some feature scores. We address this in more details when explaining features in the next section.

3.1.2 Feature description

We chose some features to improve effectiveness of our approach. All of them were normalized to lie between 0 and1:

Tweet popularity: this feature estimates the popularity of tweet t in T_q. We made the assumption that a tweet is popular if we find the same content in many other tweets. The similarity between a pair of tweets is calculated using the Lucene similarity function[5]. The score is obtained by summing the similarity score of the current tweet with all tweets in Tq

Tweet length: instinctively, the longer a sentence is, the more information it contains. We calculate this feature by counting the number of words in the tweet.

Exact term matching: this feature promotes tweets that contain terms of the topic q.

URL presence: by sharing an URL, an author would confirm the information published in his tweet or draw the attention of his followers to contents on the web. Thus, we believe that it could indicate informativeness.

URL popularity: this feature aims at measuring how important the URLs published in tweet t are in C_q. It is calculated by evaluating the frequency of URL in C_q.

Hashtag popularity: It is evaluated by counting the frequency of a hashtag h existing in a tweet t in the corpus C_q.

Topic as hashtags: this feature calculates the number of terms of topic q that are present as hashtag in tweet t.

Number of tweets: the purpose of this feature is to promote tweets published by active authors compared to tweets published by someone less active.

Mention: the more an author has been mentioned, the more important he/she is.

3.1.3 Learning approach

We exploited learning algorithms to select relevant tweets from all the Lucene outcome tweets. We used results corresponding to topics in TREC microblog 2011 as a learning set. We made 4 learning models given these 2 parameters: including the URL titles in the Lucene score or not, and scoring with BM25 or SVM. The 4 learning set were created as follow: the top 1,500 tweets

[4] http://lucene.apache.org/

[5] http://lucene.apache.org/core/3_6_1/scoring.html

resulting from each topic were obtained using the 4 configuration of Lucene. These tweets were processed to calculate the feature scores. Then, relevant and irrelevant tweets were identified since we have qrels of the 2011 track. In the four cases, we obtained an unbalanced relevance class learning distribution (2% of relevant and 98% of irrelevant). Thus, we applied an under sampling approach to reduce the number of irrelevant samples. The learning sets are then composed of the same number of relevant and irrelevant tweets. Before using these sets for learning, we used them to select the best learning algorithm. We learned and cross validated some learning approaches, and we found that the meta classifier Bagging using the classifier REPTree have the best effectiveness (i.e., Bagging: 85% of instances are classified correctly [6]). Practically, the same results were obtained given the different 4 test sets. Since we chose the learning approach and we had the set, we created the 4 learning models. Using the 2012 topics, we then created 4 runs: IRITfdvsm (TEXT field with VSM), RITfdvsmurl (TEXT and URLTITLES fields with VSM), IRITfdbm25 (TEXT field with BM25), and IRITfdBM25url (text and urltitle fields with BM25) corresponding to our 4 learning models. Only tweets classified as relevant were kept.
Participants of the adhoc task should provide as results runs containing relevant tweets to each topic, ranked given their relevance scores (not in reverse chronological order as last year's track). In our runs, we used the effectiveness classification scores to rank results.

3.2 Results and discussion

We aimed at evaluating two hypothesis: On one hand, which model between VSM and BM25 has better effectiveness? On the other hand, are the URL titles improve relevance? We had the possibility to submit only 2 runs among our 4 runs since there are 2 participants in our team. We chose to send runs using the search model having the best recall on TREC 2011 topics (0.5458 for the run using BM25 and 0.6777 for the run using VSM). Thus, we sent only runs using VSM (i.e., VSM with and without using urltitles).

	p@10	p@20	p@30	p@100	MAP	R-Precision
IRITfdvsm*	0.1915	0.1398	0.1311	0.0773	0.0886	0.1207
IRITfdbm25	0.1661	0.1449	0.1271	0.0668	0.0865	0.1216
IRITfdvsmurl*	0.1847	0.1534	0.1390	0.0803	0.0975	0.1393
IRITfdbm25url	0.1780	0.1627	0.1418	0.0758	0.0907	0.1407
TREC median			0.1808		0.1486	0.1869

Table 2. Comparison of model configurations. * Official run

Table 2 shows results of our runs. One could observe that our runs did not yield good results. We made further experiments and we found that our selected learning approach, even the best on 2011 topics[7], did not work as expected on 2012 topics. However, one could see that runs using VSM have better MAP comparing to runs using BM25. In addition, the MAP has been improved where the URL titles are used (10% of improvement in IRITfdvsmurl comparing to IRITfdvsm and 4% in IRITfdbm25url comparing to IRITfdbm25).

Figure 2 shows the ROC curves of our official runs. One may notice that there is no noticeable difference between them.

[6] We tested also fifty learning approaches among them SVM (81%), Naive Bayes (78%).

[7] We learned and cross-validated the Bagging model on 2011 topic results and we obtained the following results: 0.2982 of MAP and 0.3619 of P@30.

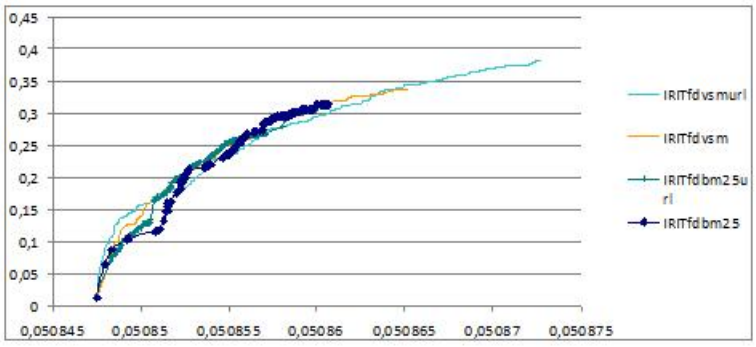

Fig. 2. ROC curves of our four runs

For next year, we will use a more appropriate learning algorithm and try to improve recall by using a query expansion technique.

References

1. Jagan Sankaranarayanan, Hanan Samet, Benjamin E. Teitler, Michael D. Lieberman, and Jon Sperling. Twitterstand: news in tweets. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '09, pages 42–51, 2009.
2. Iadh Ounis, Craig Macdonald, Jimmy Lin, and Ian Soboroff. Overview of the TREC-2011 microblog track. In *Proceeddings of the 20th Text REtrieval Conference (TREC 2011)*, 2011.
3. Jaime Teevan, Daniel Ramage, and Merredith Ringel Morris. #twittersearch: a comparison of microblog search and web search. In *Proceedings of the fourth ACM international conference on Web search and data mining*, WSDM '11, pages 35–44, New York, NY, USA, 2011. ACM.
4. Rinkesh Nagmoti, Ankur Teredesai, and Martine De Cock. Ranking approaches for microblog search. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT '10, pages 153–157, Washington, DC, USA, 2010. IEEE Computer Society.
5. Yajuan Duan, Long Jiang, Tao Qin, Ming Zhou, and Heung-Yeung Shum. An empirical study on learning to rank of tweets. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 295–303, 2010.
6. Marco Antônio Pinheiro de Cristo, Pàvel Pereira Calado, Maria de Lourdes da Silveira, Ilmério Silva, Richard Muntz, and Berthier Ribeiro-Neto. Bayesian belief networks for ir. *International Journal of Approximate Reasoning*, 34(2-3):163 – 179, 2003. Soft Computing Applications to Intelligent Information Retrieval on the Internet.
7. Lamjed Ben Jabeur, Lynda Tamine, and Mohand Boughanem. Intégration des facteurs temps et autorité sociale dans un modèle bayésien de recherche de tweets. In *Conférence francophone en Recherche d'Information et Applications (CORIA), Bordeaux, 21/03/12-23/03/12*, pages 301–316, 2012.