

Estimating Semantic Similarity between Expanded Query and Tweet Content for Microblog Retrieval

Zhijia Zhang, Man Lan*

Shanghai Key Laboratory of Multidimensional Information Processing
Department of Computer Science and Technology,
East China Normal University Shanghai 200241, P. R. China
51131201039@ecnu.cn, mlan@cs.ecnu.edu.cn*

Abstract

This paper reports the systems we submitted to the Microblog Track shared in TREC 2014 which focuses on ad hoc retrieval (i.e., retrieving top 1,000 relevant tweet for every given topic). To address this task, we adopted a two-stage framework, i.e., firstly, we performed query expansion (i.e., expanding relevant information using pseudo-relevance feedback and Google search engine results) to retrieve more relevant tweets, then extracted several effective semantic features (e.g., *Jansen-Shannon Distance*, *Overlap Similarity*, *Lucene Score*, etc) from retrieved results and built ranking model using supervised machine learning algorithms with the aid of these features to perform re-ranking. Our systems ranked 3th out of 21 teams.

1 Introduction

Microblog is a form of personal content sharing which derives from blog, and focuses on the social network-style interactions timely and rapidly. In the past few years, there has been a huge growth in the use of microblog platforms such as Twitter¹, Facebook². Promoted by that growth, companies and media organizations are increasingly seeking ways to mine microblog for information about the view points of certain topics (Kouloumpis et al., 2011). The first challenge of mining information from microblog is data retrieval.

The microblog track in TREC 2014 focuses on the real-time search on Twitter, namely, Temporally-Anchored Ad Hoc Retrieval. This ad hoc retrieval

¹<http://twitter.com>

²<http://facebook.com/>

task can be summarized as follows: “At time T , retrieve the most relevant tweets about an information need expressed as query Q .” There are two key points in this task: real-time and precise, which means that we should retrieve the most relevant tweets posted within a certain period of time. Thus, for each given query, participating team is required to return the top 1000 tweets ranked in decreasing order of predicted relevance score, which are published prior to (including) the query time defined by the organizers.

Twitter usually used to mine information because of its large amount of social information and timeless property. Although there are a wide variety of researches on microblog data for various data mining tasks (e.g., sentiment analysis, entity linking, semantic similarity, etc), relatively few work has done on microblog retrieval (Ferguson et al., 2012). (Mas-soudi et al., 2011) developed a language model with aid of two groups of quality indicators and proposed a dynamic query expansion model to perform microblog retrieval. (O’Connor et al., 2010; Li et al., ; Zhu et al.,) implemented simple but effective query expansion methods which much improved the performances of retrieval. Apart the common methods used in retrieval systems, some unique properties of microblog are exploited for retrieval purpose (Lau et al., 2011). For example, (Efron, 2010; Huang et al., 2010) showed the usefulness of *Hashtag#* in relevance feedback and query expansion. (Magnani et al., 2011; Nagmoti et al., 2010; Kwak et al., 2010) employed user information for retrieving conversations from microblog.

Following many previous work (Li et al., ; Qiang et al., ; Zhu et al., ; Hasanain et al., ; Han et al., 2012; Zhu et al., 2012; Hoang Van Duc et al., 2012) on this

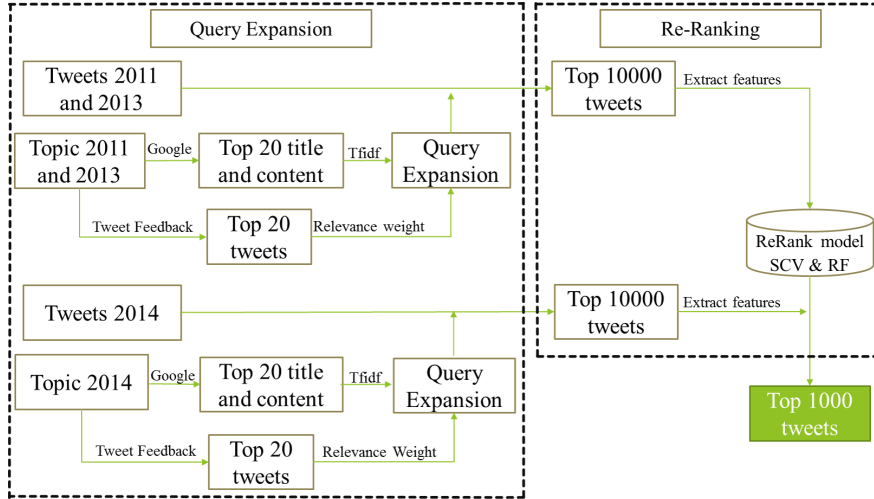


Figure 1: The framework of our proposed system.

task, we adopt a two-stage framework to address this work, i.e., query expansion (i.e., adding relevant information to query) and re-ranking (extracting several semantic features and using supervised machine learning algorithm to perform re-ranking).

The paper is organized as follows. Section 2 describes the framework of our system. Section 3 reports the experimental setting and results on training and test data. Finally, Section 4 concludes this work.

2 System Overview

In this work, we employed a two-stage framework to settle the microblog retrieval, i.e., query expansion and re-ranking.

Figure 1 depicts the architecture of our proposed system, where the left part is a query expansion module and the right part is a re-ranking module. In query expansion, we expanded the query from two distinct sources (i.e., tweet feedback and Google returned results) and optimized the best combination to perform query expansion. As for re-ranking part, we first extracted several effective semantic features (i.e., *JansenShannon Distance*, *Overlap Similarity* and *Lucence Score*, etc) from preliminary retrieval results, then we explored two ranking strategies (i.e., point-wise and pairwise) and compared several supervised machine learning algorithms with the aid of above features to perform re-ranking.

2.1 Query Expansion

Since the text length in Twitter is usually short (limited to 140 words), given a query, the relevant tweets may not contain all the words in a query. Therefore, simply using keyword matching may not be able to retrieve all relevant tweets. In order to overcome the lexical gap between tweets and query, we proposed two methods to perform query expansion. The first method aims at using tweet feedback provided by organizers to extract the relevant information from tweets. The second is to use the returned results from Google search engine to expand the query. The difference between these two methods lies on the source of relevant information, i.e., internal (tweets) and external (Google search engine).

2.1.1 Tweet Feedback

Since organizers provided a API to retrieve tweets ranked by initial *Lucene* ranking score and these top tweets are expected to be closely relevant to the search query, we proposed to expand queries with the aid of the information from these top K feedback tweets. Here we adopt a variant of $tf * idf$ to calculate the weight of each word as follows:

$$Fweight_w = \sum_{j=1}^K \left(\frac{freq(w)}{len(T_j)} \cdot score_j \right)$$

where $Fweight_w$ is the weight of word w in feedback tweets, T_j is the j th tweet returned by the API and $score_j$ is the corresponding Lucene relevant score provided by API, $freq(w)$ is the fre-

quency of word w in T_j , $len(T_j)$ is the total number of words in T_j . Since different words have quite different $Fweight_w$ scores, we performed L_1 normalization. In this way, we expanded query words by adding top P words from the top K feedback tweets according to their $Fweight_w$ scores. Obviously, this tweet feedback-based query expansion method takes both the relevancy of tweet and the frequency of word into consideration and thus it is expected to be able to comprehensively evaluate word relevance.

2.1.2 Google Returned Results

The second query expansion method is to use Google returned results. This is based on the observation that the titles and snippets of Web pages returned by Google search engine are closely related to the search query. Similar to the above tweet feedback approach, we used the top K results returned by Google search engine to expand query. To select the significant words from the results, we adopted the traditional $tf * idf$ (Joachims, 1996) to measure the relevance of the words with given query.

2.2 Re-Ranking

After the above query expansion processing, we first extracted three types of semantic relevancy features between each expanded query and tweet, and then adopted supervised learning-to-rank algorithms to perform tweet re-ranking.

2.2.1 Preprocessing Tweets

Firstly, we used the API provided by organizers to collect top 10,000 tweets for each given query. Since the collected tweet dataset contains a lot of noise, we conducted a series of preprocessing procedures, including removing non-English tweets and retweets (retweets and Non-English tweets are judged as non-relevant), removing stop words and punctuations. After that, we changed words to their lowercase and performed tokenization and stemming.

2.2.2 Features

To measure the semantic relevance between query and tweet, we used the following three types of surface text similarity measures.

Jensen-Shannon Divergence (JSD): *JSD* (Fuglede and Topsoe, 2004) is a symmetrized and

smoothed variant of Kullback-Leibler divergence, which is calculated as following:

Jensen – Shannon Divergence (S, T) =

$$\frac{1}{2}KL(P_S \parallel Q) + \frac{1}{2}KL(P_T \parallel Q)$$

$$Q(w) = \frac{1}{2}(P_S(w) + P_T(w))$$

$$KL(P \parallel Q) = \sum_{x \in X} P(x) \log \frac{P(x)}{Q(x)}$$

where $Q(w)$ denotes the distribution of the document collection of T and S , $KL(P \parallel Q)$ means the Kullback-Leibler divergence between distribution P and Q . Here we proposed four *JSD* features by calculating the *JSD* between tweet and original query in unigram and bi-gram, denoted as *JSD_TvQu* and *JSD_TvQb*, and the *JSD* between tweet and Google returned results in unigram and bi-gram, denoted as *JSD_TvGu* and *JSD_TvGb*, respectively.

Overlap Similarity (OS): *OS* is a simple and effective similarity measure and calculated as follows:

$$Overlap\ Similarity = \frac{|A \cap B|}{|A|} \text{ or } \frac{|A \cap B|}{|B|}$$

where $|A \cup B|$ denotes the size of intersection of set A and set B and $|A|$ means the size of set A . We denote the word sets collected from tweet, query and Google returned results as T , Q and G . The ratio of $|T \cap Q|$ to $|Q|$ (denoted as *OS_TvQ*) and the ratio of $|T \cap G|$ to $|G|$ (denoted as *OS_TvG*) were calculated as *OS* features.

Lucene Score (LS): *Lucene Score* is a state-of-the-art similarity score which combines several measures³, e.g., cosine formula and dirichlet prior. It is calculated as follows:

$$Lucene\ Score(q, d) = coord_{q,d} \cdot queryNorm_q \cdot$$

$$\sum_{t \text{ in } q} (tf_{t,d} \cdot idf_{t,d} \cdot weight_t \cdot norm_{t,d})$$

where $coord_{q,d}$ is a grade factor defined as $\frac{Overlap}{MaxOverlap}$, $queryNorm_q$ is a normalizing factor which does not affect the ranking, $tf_{t,d}$ denotes the term frequency in d , $idf_{t,d}$ is the inverse document frequency, $weight_t$ is the weight of term t in query q and $norm_{t,d}$ is encapsulated by a small part of $weight_t$ and the length of document d . The official API provides this *Lucene Score* between query and returned tweets.

³<http://lucene.apache.org/core/2.9.4/api/core/org/apache/lucene/search/Similarity.html>

2.2.3 Learning-to-rank

We used supervised learning-to-rank approach to perform re-ranking. Generally, the learning-to-rank approach can be divided into three groups: point-wise, pair-wise and list-wise. In this work, we adopted the first two strategies.

In point-wise strategy, given a query, for each returned tweet, we first built a *query-tweet* pair with its relevant label. Then we adopted regression algorithm to build a regression model. For each unlabeled *query-tweet* pair, we used the regression model to calculate its relevant score. In this work we employed two regression algorithms, i.e., *SVR* and *KNeighbors Rgressor*.

In pair-wise strategy, given a query and several returned tweets, we constructed many *tweet1-tweet2* pairs. If *tweet1* is more relevant than *tweet2* in terms of the given query, this pair label is set as *1*, otherwise *0*. By doing so, we adopted supervised classification method to predict the label of each pair. Since the classification model provides a confidence score for each instance, then these tweets are ranked by this confidence score.

3 Experiments

3.1 Datasets

The organizers provided two data sets of microblog track with gold annotation in TREC 2011 and TREC 2013. Therefore, we experimented on these two training datasets and employed the configured systems on TREC 2014.

In the microblog track of TREC 2014, the participants could interact with a tweet collection stored remotely via a search API. The motivation for this evaluation-as-a-service design is to increase the size of the collection while adhering to Twitter's terms of service. Past arrangements allowed teams to acquire local copies of a canonical corpus. But the logistics of this approach prohibited scaling the corpus size up dramatically. The total amount of tweets we obtained from the API is about 243 million which gathered from the public Twitter stream from February 1, to March 31, 2013 (inclusive).

The tweets downloaded from API are in JSON format, which contain many fields, i.e., *id* (the unique identifier assigned by twitter), *rsv* (the relevance score of tweet), *text* (text of tweet),

retweeted_count (the number of times this tweet has been retweeted) etc. Since the raw text of each tweet contains a lot of noise, we performed data preprocessing to remove non-English tweets and retweets. We also used Natural Language Toolkit⁴ for tokenization, stemming and removing stop words.

Totally for each query, we collect up to 10,000 tweets from API ranked by Lucene score as our original datasets.

3.2 Performance Evaluation

The evaluation metrics provided by organizer are *Mean Average Precision (MAP)*, *Precision at rank 30 (P@30)* and *Reciprocal Precision (R-Prec)* which are defined as follows.

MAP: Mean average precision depends on the precision at each point when a new relevant tweet gets retrieved which is calculated as:

$$MAP = \frac{1}{N} \cdot \sum_{j=1}^N \frac{1}{Q_j} \cdot \sum_{i=1}^{Q_j} P(T_i)$$

where Q_j denotes the number of relevant tweets for query j , N presents the number of the queries and $P(T_i)$ is the precision of the relevant tweets in top i tweets. *MAP* provides a single-figure measure of quality across recall levels.

P@30: This metric denotes that the precision of top 30 tweets. It has advantage of not requiring any estimate of the size of the set of relevant tweets and the total number of relevant tweets for query has a strong influence on *P@30*.

R-Prec: If there are $|Rel|$ relevant tweets for a query, we examine the top $|Rel|$ tweets returned by the system, and find that r tweets are relevant, then the *R-Prec* is calculated as $r/|Rel|$. Specifically, the recall of this tweets set is also $r/|Rel|$.

In these three evaluation metrics, *MAP* is the main evaluation criteria for this task.

3.3 Preliminary Experiments on Training Data

In order to determine the optimum system configuration, we conducted preliminary experiments on the datasets of TREC 2011 and TREC 2013. To make reasonable comparison, we performed cross-over experiments on the two datasets.

⁴<http://www.nltk.org/>

3.3.1 Query Expansion

Due to the shortness of information of initial query provided by organizer, we performed several query expansion methods with the aid of original query, tweet feedback and Google returned results.

Only Query (Q):

We performed API search using the initial query Q provided by organizer only. This serves as our baseline for query expansion. The weight of each word in Q is calculated as the ratio of corresponding word frequency in query to the total number of words in query.

Query + Tweet Feedback (Q+T):

This query expansion pattern consists of the initial query and the top 20 relevant words extracted from the top 20 feedback tweets (T). The weight of each word in this query pattern is the sum of respective word weight in two set (If word w not exist in set, the weight of w is 0 in corresponding set).

Tweet Feedback + Google Returned (T+G):

We assumed that the top 20 Google returned results are quite relevant to the current query. Besides, we extracted the top 20 relevant words from these top 20 Google returned results as set G , then built a new expansion pattern consisted of T and G . Towards the weight of each term, the calculation is the same as before.

Query + Tweet Feedback + Google Returned (Q+T+G):

We combined all elements mentioned before (i.e., initial query, the top 20 relevant words in top 20 tweets of feedback results and the most relevant 20 words in top 20 documents of Google returned results) as a new query expansion pattern. For the weight of each term, the computational method is similar as before but three sets rather than two sets.

To compare the performance of different query expansion patterns, we used the top 1,000 tweets returned by API.

Table 1 shows the results of different query expansion methods on two TREC training datasets. We find that the query expansion pattern by combining tweets feedback with Google returned results outperformed other query expansion patterns. Based on

Datasets	TREC 2011		TREC 2013	
	MAP(%)	P@30	MAP(%)	P@30
Q	25.68	30.00	20.44	37.56
Q+T	41.17	45.99	28.30	46.50
T+G	46.84	50.07	32.99	52.89
Q+T+G	44.70	50.02	32.01	52.00

Table 1: Performances of different query expansion methods on two training datasets.

Dataset	2011		2013	
	MAP(%)	P@30	MAP(%)	P@30
T+G(1:1)	46.84	50.07	32.99	52.89
T+G(2:1)	45.93	50.02	32.75	53.22
T+G(3:1)	47.34	50.34	33.02	53.56
T+G(5:1)	47.11	50.00	33.00	53.72

Table 2: Results of different ratio of T and G on two training datasets.

our observation of the words in the three sets Q , T , G , we find that the words appearing in Q also have high weights in the other two sets. However, the words in T and G are quite different. That means, the words from the two query expansion methods may make up for their shortage of information and improve the performance. Thus, for the following experiments, we adopted the $T+G$ pattern to perform query expansion.

Furthermore, in order to examine the different contributions made from the two components (i.e., tweet feedback and Google returned results), we also conducted further experiments on different ratios between these two sets. Table 2 lists the further experimental results with different ratios between T and G . Since the query expansion pattern of tweets feedback and Google returned results with the ratio of 3 : 1 performed best, for the follow-up experiments, we used the returned tweets by API with this setting as datasets.

3.3.2 Re-ranking

We adopted two learning-to-rank strategies to perform re-ranking. For point-wise strategy, we adopted a *rbf*-based *SVR* algorithm with $c = 1$ and a *Kneighbors Regressor* algorithm with $K = 1$. For pair-wise strategy, the *Random Forest* and *Rank-Boost* algorithms with default parameters were used.

Table 3 shows the results of re-ranking part with different strategies and algorithms. From Table 3,

Strategy	Point-wise				Pair-wise			
	SVR		Kneighors Regressor		Random Forest		RankNet	
Algorithm	MAP	P@30	MAP	P@30	MAP	P@30	MAP	P@30
2011	50.29	53.33	37.73	43.67	50.74	52.65	47.15	48.71
2013	34.70	55.22	28.60	50.44	36.12	56.39	32.60	53.33

Table 3: Results of re-ranking part on two training datasets.

it is interesting to see that the pair-wise strategy method generally outperformed the point-wise strategy method in terms of *MAP* measure. In terms of *P@30* measure, the performance of *SVR* is close to that of *Random Forest*. Thus, for the final systems, we selected these two algorithms with corresponding parameters to perform re-ranking.

3.4 System Configuration

Based on the preliminary experiments on training data, we chose the setting of tweets feedback and Google returned results with the ratio of 3 : 1 for query expansion part.

According to the results of re-ranking part, we adopted a *rbf*-based *SVR* algorithm with $c = 1$ implemented in scikit-learn toolkit⁵ and *Random Forest* algorithm with default parameters implemented in RankLib⁶. With regard to the setting in RankLib toolkit, the metric is set as *MAP* and the iteration is set as 300 times.

Besides, we find that the performance of using TREC 2013 as training data is much better than that of using TREC 2011. However, since the dataset of TREC 2013 contains more relevant tweets (9,000 relevant tweets) than that of TREC 2011 (3,000 relevant tweets), we also consider to build model on TREC 2013 only.

In summary, the configurations of our four submissions are shown in Table 4.

System	TrainData	F_G Ratio	Re-rank Strategy	Algorithm
SVRALL	2011,2013	3:1	Pointwise	SVR(kernel=rbf,c=1)
SVR2013	2013	3:1	Pointwise	SVR(kernel=rbf,c=1)
RFALL	2011,2013	3:1	Pairwise	RF(para=default)
RF2013	2013	3:1	Pairwise	RF(para=default)

Table 4: Configurations of our four submissions. *RF* stands for *Random Forest* algorithm.

⁵<http://scikit-learn.org/stable/>

⁶<http://people.cs.umass.edu/vdang/ranklib.html>

System	MAP(%)	P@30(%)	R-Prec(%)
SVRA	53.51	70.00	53.30
SVR13	53.86	70.61	53.57
RFA(3)	55.29(3)	71.33(2)	54.27(5)
RF13	55.19	71.15	54.88
PKUICST3(1)	58.63(1)	72.24(1)	57.27(1)
hltcoe3(2)	57.07(2)	71.21(3)	56.60(2)
Best	67.51	83.45	66.59
Median	41.55	62.61	43.95

Table 5: Results of our four submissions, the top2 ranked systems, the best and median synthetic aggregated runs released by TREC official for the ad hoc search task. The numbers in the brackets are the rankings of corresponding metric.

3.5 Results and Discussion

Table 5 shows the official released results of our submitted four runs with the top 2 ranked systems and the best and median synthetic aggregated runs released by TREC official for ad hoc task.

Form Table 5, it is interesting to find following observations.

Firstly, the pair-wise strategy methods outperform point-wise strategy method. This consequence is consistent with the results on training data, which indicates that pair-wise is more applicable to this re-ranking task than point-wise. Furthermore, the regression model for point-wise strategy costs vast amounts of training time. Thus, we prefer to employ pair-wise strategy.

Secondly, the TREC 2013 dataset only is quite general to build the model. The performance of different datasets is almost equal in terms of the same re-rank strategy. There are almost no significantly improvement with the addition of TREC 2011. Moreover, the time cost of building model on TREC 2013 is much lower than which on TREC 2011 with TREC 2013.

In summary, the best result of our submissions ranked 3th out of 21 teams. Comparing with the

top2 ranked systems, our systems are promising, although the simple features were adopted.

4 Conclusion

To address TREC 2014 Microblog track, we explored a two-stage framework, i.e., query expansion and re-ranking. For query expansion, the combination of two methods based on different sources of relevant information, i.e., internal (tweets) and external (Google search engine), outperformed each method alone. To re-rank tweets, the pair-wise strategy outperformed the point-wise strategy.

In future work, we would investigate more effective query expansion methods. Also we will explore more features, such as *URL* information, to enrich the feature set and improve the performance of re-ranking.

Acknowledgments

This research is supported by grants from National Natural Science Foundation of China (No.60903093), Shanghai Collaborative Innovation Center of Trustworthy Software for Internet of Things (ZF1213), and the Science and Technology Commission of Shanghai Municipality under research grant no. 14DZ2260800.

References

- Miles Efron. 2010. Hashtag retrieval in a microblogging environment. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 787–788. ACM.
- Paul Ferguson, Neil OHare, James Lanagan, Owen Phelan, and Kevin McCarthy. 2012. An investigation of term weighting approaches for microblog retrieval. In *Advances in Information Retrieval*, pages 552–555. Springer.
- Bent Fuglede and Flemming Topsøe. 2004. Jensen-shannon divergence and hilbert space embedding. In *IEEE International Symposium on Information Theory*, pages 31–31.
- Zhongyuan Han, Xuwei Li, Muyun Yang, Haoliang Qi, Sheng Li, and Tiejun Zhao. 2012. Hit at trec 2012 microblog track. In *Proceedings of Text REtrieval Conference*.
- Maram Hasanain, Latifa Al-Marri, and Tamer Elsayed. Qu at trec-2013: Expansion experiments for microblog ad hoc search.
- Thong Hoang Van Duc, Thomas Demeester, Johannes Deleu, Piet Demeester, and Chris Develder. 2012. Ugent participation in the microblog track 2012. In *Text Retrieval Conference (TREC-2012)*, pages 1–5.
- Jeff Huang, Katherine M Thornton, and Efthimis N Efthimiadis. 2010. Conversational tagging in twitter. In *Proceedings of the 21st ACM conference on Hypertext and hypermedia*, pages 173–178. ACM.
- Thorsten Joachims. 1996. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Technical report, DTIC Document.
- Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. 2011. Twitter sentiment analysis: The good the bad and the omg! *ICWSM*, 11:538–541.
- Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. 2010. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pages 591–600. ACM.
- Cher Han Lau, YueFeng Li, and Dian Tjondronegoro. 2011. Microblog retrieval using topical features and query expansion. In *TREC*. Citeseer.
- Cheng Li, Yue Wang, and Qiaozhu Mei. A user-in-the-loop process for investigational search: Foreseer in trec 2013 microblog track.
- Matteo Magnani, Danilo Montesi, Gabriele Nunziante, and Luca Rossi. 2011. Conversation retrieval from twitter. In *Advances in Information Retrieval*, pages 780–783. Springer.
- Kamran Massoudi, Manos Tsagkias, Maarten de Rijke, and Wouter Weerkamp. 2011. Incorporating query expansion and quality indicators in searching microblog posts. In *Advances in Information Retrieval*, pages 362–367. Springer.
- Rinkesh Nagmoti, Ankur Teredesai, and Martine De Cock. 2010. Ranking approaches for microblog search. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, volume 1, pages 153–157. IEEE.
- Brendan O’Connor, Michel Krieger, and David Ahn. 2010. Tweetmotif: Exploratory search and topic summarization for twitter. In *ICWSM*.
- Runwei Qiang, Yue Fei, Yihong Hong, and Jianwu Yang. Pkuicst at trec 2013 microblog track.
- Siming Zhu, Zhe Gao, Yajing Yuan, Hui Wang, and Guang Chen. Pris at trec 2013 microblog track.
- Bolong Zhu, Jinghua Gao, Xiao Han, Cunhui Shi, Shenghua Liu, Yue Liu, and Xueqi Cheng. 2012. Ictnet at microblog track trec 2012. Technical report, DTIC Document.