PKU_ICST at TREC 2011 Microblog Track

Feng Liang, Runwei Qiang, Jianwu Yang*

Institute of Computer Science and Technology, Peking University

{liangfeng, qiangrunwei, yangjw}@pku.edu.cn

Abstract

This paper describes the PKU_ICST participation in the TREC 2011 Microblog track. In the first year of Microblog track, we designed a group of experiments to verify whether external resources and future resources would improve the performance of our system. Moreover, given that microblog track is a real-time adhoc task, we explored an approach making use of the temporal evidence. To obtain a better performance, we employed different strategies to generate final results.

1. Introduction

In this paper, we describe our entry into the TREC 2011 Microblog track. This year, Microblog track focuses on a realtime search task whereby a user's information need will be represented by a query at a specific time. In particular, user wishes to see the most recent but relevant information to the query. Hence, the system should answer a query by providing a list of relevant tweets ordered from latest to earliest, starting from the time the query was issued.

In the first run of Microblog track, we employed different methods to verify whether external resources and future resources would help our system improve its performance. Then, we incorporated temporal evidence into baseline method to meet the real-time demand. Lastly, we explored both static and dynamic methods to choose the final result.

2. Microblog Realtime Adhoc Task

In this chapter, we describe our approach for the Realtime Adhoc Task in detail.

2.1. System Overview

Our system architecture is shown in Figure 1. We dealt with corpus and query in parallel. For Tweet2011 Corpus, we did necessary preprocessing including non-English word filtering and retweet elimination. In the next step, we employed two optional methods, including adding outer link's information to original corpus, and processing the hashtag information to original corpus, to generate three candidate corpuses to be retrieved. For the test query, we explored four approaches, namely retrieving NYTimes (New York Times) news articles, making use of WordNet relationship, combining both NYTimes and WordNet resource, selecting top tweet as relevance feedback, to expand the original query. In addition, both stemming and stop words elimination were applied to corpus and query processing. After the retrieval step with the help of Lemur1 IR[1] toolkit, a relevance score is assigned to each tweet. Next, we incorporated the temporal evidence into the normalized score. Finally, we developed two strategies to select results and re-ranked the result list according to tweet id.

* *corresponding author*

[1] *http://www.Lemurproject.org/Lemur.php*

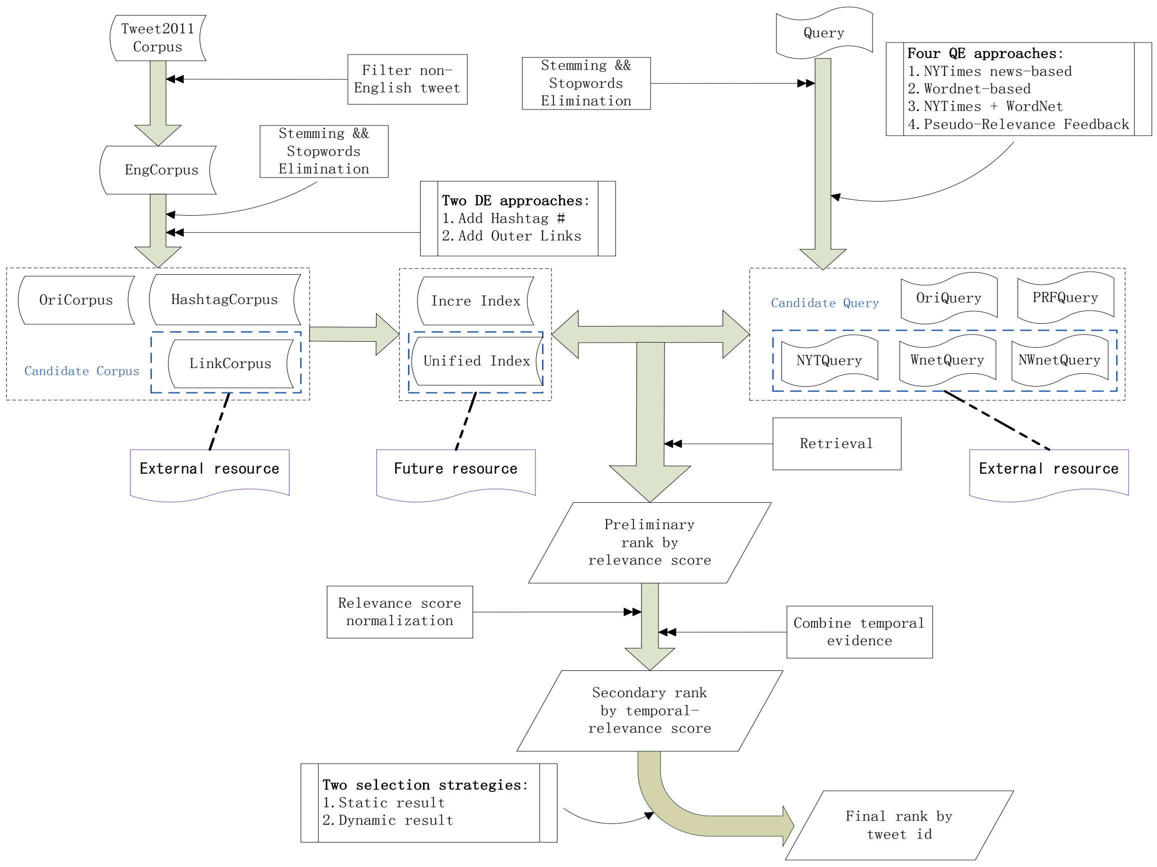


***Figure 1**: System Architecture*

2.2. Preprocessing

The size of the corpus is approximately 16 million tweets over a period of 2 weeks. We crawled HTML version corpus directly from Twitter, using a provided tool.
In the preprocessing step, we dealt with Tweet2011 Corpus in following ways:

- **Non-English tweets filter:**

we filtered out all tweets that have words encoded with non-ASCII code.

- **Simple retweet elimination:**

we eliminated tweets that begin with 'RT' with the consideration that these tweets are simple retweets without any other additional information.

2.3. Retrieval Models

In the baseline stage, we employed four baselines just using original query as follows:

- **Boolean OR Model as baseline1**

System returned the most recent 1000 tweets that contain any of the query terms.

- **Probability OR Model as baseline2**

System firstly calculated the probability of a given query, using Indri Retrieval Model [1] which is based on a combination of the language modeling and inference network. We estimated the probability by the following formula:

$$P(query) = 1 - \left(\prod_{i=1}^{n}\left(1 - P(term_i)\right)\right) \qquad (1)$$

Where $p(term_i)$ is the probability of a query term.
For instance, query MB004 is represented by Indri query language as follows:

```
<query>
#or(Mexico drug war)
</query>
```

After obtaining a list of tweets with relevance scores, we selected top *Res_Num* tweets and re-ranked them according to their tweet ids. Note that *Res_Num* is the parameter discussed in section 2.7.

- **Probability OR Model with the addition of simple rules as baseline3**

In baseline3, we incorporated following two simple rules into baseline2:
Rule1: contents within Quotation marks must be conjunctive, and this part should be assigned to a higher weight, i.e. 2.5. For example, query MB014 is represented by Indri language as follows:

```
<query>
1.0 #or(release of the rite)
2.5 #1(the rite)
</query>
```

Rule2: conjunctive capitalized words must be contained within a fixed-length, i.e. 5, text window, and this part should receive a higher weight, i.e. 2.0. For instance, query MB001 is represented by Indri language as follows:

```
<query>
1.0 #or(bbc world service staff cuts)
2.0 #uw5(bbc world service)
</query>
```

- **Language Model using KL-divergence method as baseline4**

We employed language model [2] choosing KL-divergence method with the help of Lemur toolkit, to estimate the likelihood of a given query and tweets. To avoid the zero-probability problem and overcome data sparsity in Twitter in which most tweets usually contain a few words, we applied different smoothing methods [3], namely Jelinek-Mercer, Dirichlet prior, and Absolute discounting.
As we did in baseline2, we then selected top *Res_Num* tweets and re-ranked them based on their tweet id, from latest to earliest.

2.4. Future Resource vs. Present Resource

We obtained three corpuses, i.e. oriCorpus, hashtagCorpus and linkCorpus, after the document expansion approach discussed in section 2.5.2. Then, we built a unified index as well as an incremental index on all corpuses.
Unified index is considered as a future resource because it uses the information that would not have been available to the system at the timestamp of the query, while incremental index is a present resource for it only takes advantages of tweets available at the time the query is issued.

2.5. External Resource vs. Local Resource

In this year's track, external resources, such as NYTime news articles and WordNet, were used in the query expansion and document expansion process. Also, we took advantages of local resources to improve system's performance.

2.5.1. Query Expansion (QE)

We explored four query expansion approaches, namely NYTimes news-based QE, WordNet-based QE, mixture QE of NYTimes news and WordNet, and pseudo-relevance feedback QE, of which the first three methods made use of external resource, while the pseudo-relevance feedback QE approach expanded the original query by making the most of local resource.

● **NYTimes news-based QE:** As news article and microblog are both time-sensitive, it not hard to consider of making use of news articles as external resources to expand original query [4]. In order to obtain a parallel news corpus, we chose New York Times as our external resource of news articles. We crawled news articles through NYTimes's advanced search page[2] observing the following steps:

1) Set up the date ranging from 24th January 2011 to 8th February.
2) Treat the whole test query as the news search query.
3) Only keep nouns, capitalized words, and phrases in within Quotation marks as the news search query.
4) Only keep person's name, place name, capitalized words and words indicating time as the news search query.

Note that at the end of step 2 to 4, we checked whether the number of retrieved news articles is greater than 10, if so, we stopped the crawling process. Otherwise at the end of step 4, the number of retrieved news articles is less than 10, we then did not expand this query.

Term relationships were extracted from the articles we crawled from NYTimes search page. We used a co-occurrence analysis, which assumes that terms that often co-occur in the news articles are related. In particular, we consider a text window of fixed size, i.e. terms whose distance is not larger than that size are considered to co-occur, and this add 1 into the global count of their co-occurrences. To filter out noise, we used a threshold of 10 on the global count, i.e. we only retained the co-occurrences whose count is at least 10. For each query term, we retained a set of strongest co-occurrences as its expanded words. Note that the windows size and the number of expanded words are set up as parameters.

● **WordNet-based QE:** We just considered of expanding noun query terms by using their synonyms relationship in WordNet. For each query term, we simply selected the strongest synonyms as its expanded words. In particular, we kept a global count for each query term, and for each sense we added every word's count which indicates how often this word appears in the specific sense into the global count. From a set of candidate words, we chose 10 strongest words as query term's expanded words.

● **Mixture QE of NYTimes news and WordNet:** In this mixture query expansion approach, we retained 10 terms as expanded words, of which 5 words were selected from 5 strongest co-occurrences in NYTimes news-based QE while the other 5 words were

[2] *http://query.nytimes.com/search/advanced?srchst=m*

obtained based on WordNet-based QE.

- **Pseudo-Relevance feedback QE:** Pseudo-relevance feedback [5] is widely used in query expansion process. In general, pseudo-relevance process obtains a subset of documents to create a new query with the assumption that top retrieved documents are relevance. In our pseudo-relevance approach, however, we retrieved the first tweet from the subset composed of the most recent 100 tweets and rebuilt the query by adding tweet's content into the original query for the following reasons:

1) Given that twitter posts are updated fast, the tweet which is close to query's timestamp could represent user's information better. That's why we chose the most recent 100 tweets as the candidate pseudo-relevance document.

2) Our previous experiments based on the 12 example queries showed that retrieval results were sensitive to the number of words added into the original query. That's why we combined only one tweet's content with the original query.

Notice that we ranked tweets by the probability that the given query could be generated by the tweet language model. We estimated this probability by calculating the unigram language model as follows:

$$P(Q|T) = \prod_{i=1}^{n} P(q_i \mid T) \tag{2}$$

$$P(q_i \mid T) = (1-\lambda)\frac{f_{q_i,T}}{|T|} + \lambda\frac{C_{q_i}}{|C|} \tag{3}$$

Where $f_{q_i,T}$ is the number of times a query term occurs in the tweet, C_{q_i} is the number of times a query term occurs in the collection of the most recent 100 tweets.

2.5.2. Document Expansion (DE)

Given that the length of tweet is limited by 140 characters, it is necessary to expand the document, i.e. tweet to overcome the vocabulary-mismatch problem. In this year's run, external resources were used in document expansion approach by add the content extracted from title field of outer link. Also, we made use of local resources to generate expanded terms.

- **Outer link DE:** We extracted outer links from tweets as an external resource to expand document, i.e. tweet. For each outer link, we extracted the content wrapped in title tag by analyzing html code. Then we mixed original tweet with the extracted words as the new document. Note that in this year's track, we have not downloaded the whole collection of outer links because of Great Fire Wall in China.
- **Hashtag DE:** Hashtag plays an import role in Twitter, signaling aspects of a tweet's meaning such as its topic or its intended audience [6]. For instance, *#bbcworldservicecut* ostensibly marks tweets related to affair about BBC world service staff cut. We created a lexicon by collecting all terms in 50 queries except stop words. When processing a tweet, we put a term into the original document if it is a sub-string of the hashtag with the tweet. For example, if a tweet has a hashtag *#bbcworldservicecut*, then we will put BBC, world, service and cut into it.

2.6. Temporal Evidence

Considering that the system prefers latest tweets, we should normalize the score by taking temporal factor into consideration:

$$Score_{temp}(tweet\_id) = func(tweet\_id) \times Score_{pre}(tweet\_id) \qquad (4)$$

Where $Score_{pre}(tweet\_id)$ is the preliminary score and $func(tweet\_id)$ is a decreasing quadratic function, which penalizes tweets that are far away from the time query is issued smoothly. We suppose that the temporal factor ranges from α to 1. Let pf denote the penalty factor which equals to $1-\alpha$. Therefore $func(tweet\_id)$ should be computed as follows:

$$\begin{cases} func(tweet\_id) = -k \cdot (tweet\_id - MAX_{tweet\_id})^2 + 1 \\ k = \frac{(1-\alpha)}{(MIN_{tweet\_id} - MAX_{tweet\_id})^2} = \frac{pf}{(MIN_{tweet\_id} - MAX_{tweet\_id})^2} \end{cases} \qquad (5)$$

Where $pf \in (0,1)$

As shown in Eq.5, a higher pf will increase the impact of time, so that the latest tweet will be assigned a higher weight.

2.7. Generate Final Results

Two strategies are used to generate final results from the scores which have been revised by the temporal evidence.

- **Static strategy**

Simply choose Res_Num highest-score results as the final results. Note that Res_Num is a parameter to be tuned.

- **Dynamic strategy**

To begin with, we normalized the score ranging from 0 to M, which was an integer tuned from 1 to 20. Different from the static strategy, we suppose that the number of final result should be dynamic since the number of the relevant tweets varies from query to query. So we propose a tricky method to select final results dynamically.

1) Cut the scores into (M+1) intervals: [0, 1), [1, 2), … , [M, M+1)

2) Count the number of scores in each interval: $n_{0,1}, n_{1,2}, \dots, n_{M,M+1}$

3) Compute the absolute difference between each adjacent number such as $n_{0,1}$ *and* $n_{1,2}$. Then find the maximum difference. Suppose that $n_{t-1,t}$ and $n_{t,t+1}$ make the maximum absolute difference.

4) Any result whose score is higher than t will be chosen to the final results list.

Finally, we sort the final results list according to the tweet id.

3. Result Analysis

In this section, we analyze the result of our approaches. In this year's track, all 184 submitted runs were pooled to depth 30 according to the retrieval scores indicated in each run. Among 50 topics, query 50 did not have any relevant tweets in the pool, and only 33 topics have tweets judged to be highly relevant. Thus, the evaluation includes two separate sets of scores, first of which considers all relevant and highly relevant tweets as relevant and is over 49 topics, while the second considers only highly relevant tweets, and is over 33 topics.

3.1. Analysis of official runs

Table 1 and Table 2 show the ***allrel*** and ***highrel*** performance values of our submitted four runs.

Run_ID	P@30	MAP	R-Prec
PKUICST	0.3796	**0.2757**	**0.3492**
PKUICST2	**0.3905**	0.2221	0.2725
PKUICST3	0.2830	0.2478	0.3134
PKUICST4	0.2177	0.2207	0.2490

***Table 1**: The allrel performance of our submitted runs*

Run_ID	P@30	MAP	R-Prec
PKUICST	0.1394	**0.2555**	**0.2637**
PKUICST2	**0.1414**	0.2380	0.2563
PKUICST3	0.0980	0.2273	0.2262
PKUICST4	0.0848	0.2126	0.1931

***Table 2**: The highrel performance of our submitted runs*

All submitted runs adopted pseudo-relevance feedback query expansion method, original corpus with incremental index and language model using KL-divergence retrieval model. Furthermore, temporal evidence has been integrated into relevance score. The only difference among the four submitted runs is the result selection strategy. In particular, the PKUICST run applied dynamic strategy to generate its result list, while the PKUICST2, PKUICST3, PKUICST4 applied static strategy with the parameter *Res_Num* set as 30, 100, 300 respectively.

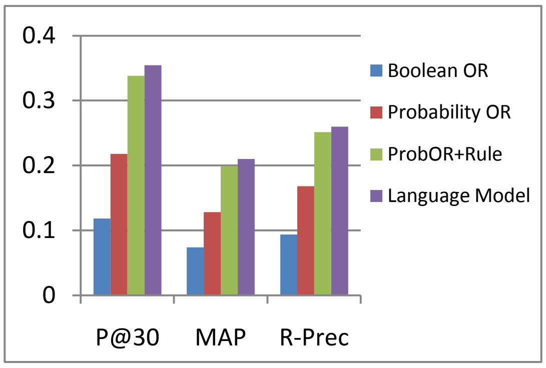
From the evaluation result, we can see that the run with dynamic result number, i.e. PKUICST achieved a good P@30 value and better MAP and R-Prec values, compared with the runs that have static result number, which indicates that the dynamic strategy performs quite stably. Besides, PKUICST2's value of P@30 outperformed others. Given that the average number of dynamic result is about 75, we could infer that the less the result number is, the better the retrieval result is.

3.2. Analysis of unofficial runs

In addition to the submitted runs, we also did some complementary experiments for comparison. Notice that all the results are produced after parameter tuning.

3.2.1. Model Comparison

To compare different models, we employed baseline1 to baseline 4, utilizing the original query and corpus with incremental index. Both ***allrel*** (on the left side) and ***highrel*** (on the right side) evaluation results are shown in Figure 2.



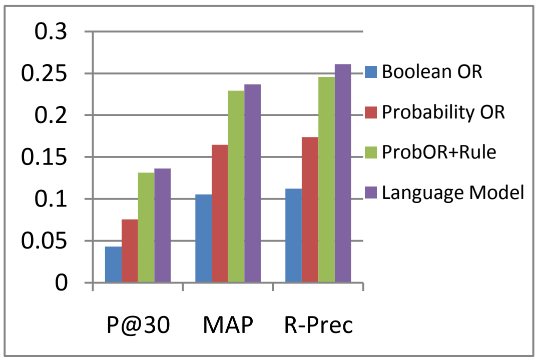


***Figure 2:** results of model comparison*

From figure 2, we can find that among four baselines, language model using KL-divergence outperforms other models.

3.2.2. Future Resource vs. Present Resource

We compared the results that original query runs on original corpus with different kinds of index. As shown in Figure 3, we could figure out that unified index which takes advantages of future resources yields a better performance than incremental index.

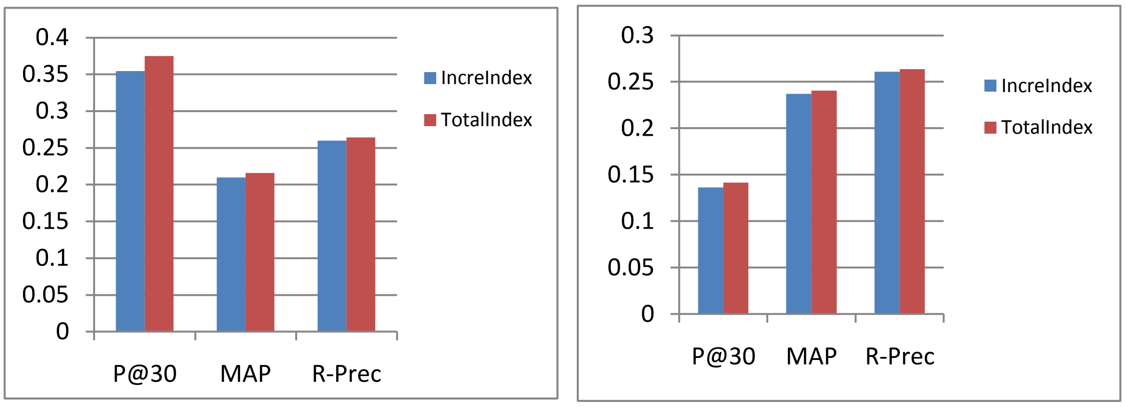


Figure 3: results of index comparison

3.2.3. External Resource vs. Local Resource

- **Query Expansion Comparison**

In section 2.4, we describe our four query expansion approaches and the results of different query expansion comparison are present in Figure 4. As shown in Figure 4, we could see that first three query expansions which made use of external resources did not increase the performance of system, compared with original query without any query expansion. However, pseudo-relevance feedback approach which only utilized the local resource yields a better performance, increasing the P@30 value by 8%.

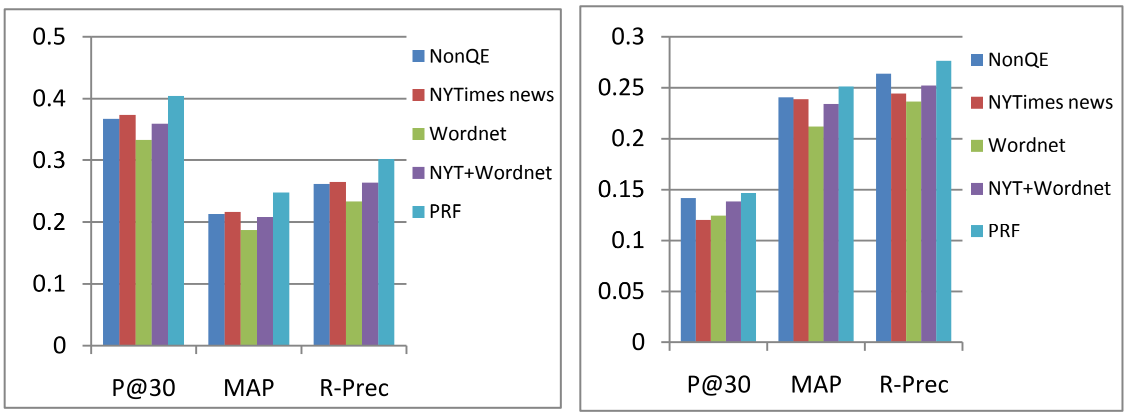


Figure 4: results of query expansion comparison

- **Document Expansion Comparison**

Note that pseudo-relevance feedback query expansion approach obtain excellent evaluation values, we adopted it to our document expansion experiment. From the results shown in Figure 5, we could draw a conclusion that external resources in document expansion approach, i.e. outer links, improve the performance dramatically. However, local resources such as hashtag decrease the retrieval result slightly.

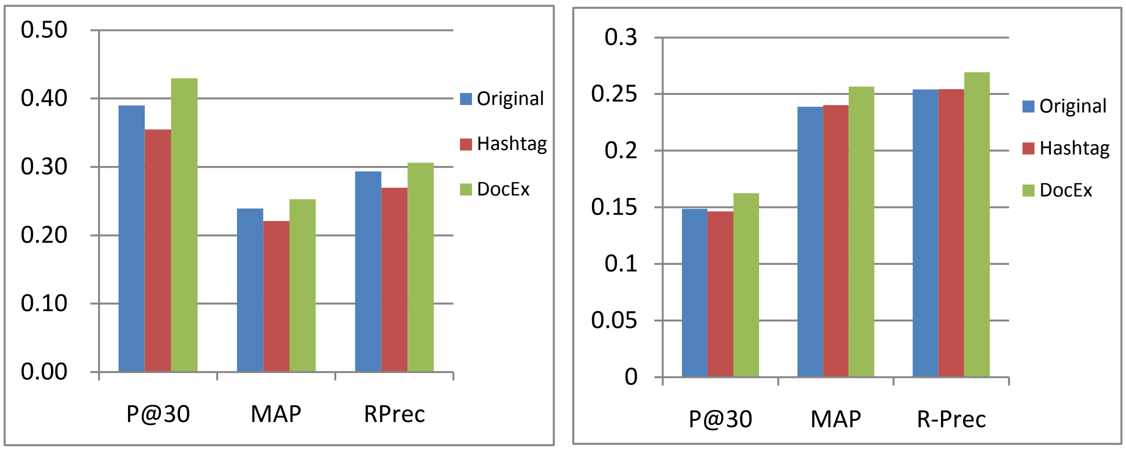


Figure 5: *results of query expansion comparison*

4. Conclusion and Future work

In this paper, we present our system for Microblog Real-Time Retrieval Task. We employed four various query expansion methods to overcome the vocabulary-mismatch problem. In addition, we developed two different approaches to expand the original document. Then, we incorporated temporal evidence into relevance score calculated by preliminary retrieval model, which made a trade-off between recentness and relevance. To generate the final result, we explored two selection strategies based on the score revised by temporal evidence and then re-ranked the result list according to tweet id. From the evaluation results, we can see that our system have achieved competitive results. In the future work, we will complement the outer link set to generate a more comprehensive corpus. Besides, we will build a more general and robust retrieval model with combination of relevance and recentness.

5. Acknowledgement

The work reported in this paper was supported by the National Natural science Foundation of China Grant 60875033.

References

[1] D. Metzler, T. Strohman, H. Turtle, and W. Croft. Indri at TREC 2004: Terabyte track. In *proceeding of the 2004 Text Retrieval Conf*, 2004.

[2] Zhai, C. and Lafferty, J. 2001. Model-based feedback in the language modeling approach to information retrieval. *CIKM 2001: Proceedings of the tenth international conference on Information and knowledge management*. (2001), 403-410.

[3] Zhai, C. and Lafferty, J. 2004. A Study of Smoothing Methods for Language Models Applied to Information Retrieval. *ACM Transactions on Information Systems*. 2, 2 (2004), 179-214.

[4] X. Zhao and J. Jiang. An empirical comparison of topics in Twitter and traditional media. *Technical Paper Series, Singapore Management University School of Information System.*

[5] Buckley, C., Salton, G., and Allan, J., Automatic Retrieval with Locality Information Using Smart, In The First Text REtrieval Conference (TREC-1), National Institute of Standards and Technology, Gaithersburg, MD, 1992, pp. 59-72.

[6] Efron, M. 2010. Hashtag retrieval in a microblogging environment. *Proceeding of the 33rd*

international ACM SIGIR conference on Research and development in information retrieval (Geneva, Switzerland, 2010), 787-788.