PKUICST at TREC 2015 Microblog Track: Query-biased Adaptive Filtering in Real-time Microblog Stream

Feifan Fan Yue Fei Chao Lv Lili Yao Jianwu Yang* Dongyan Zhao
{fanff, feiyue, lvchao, yaolili, yangjw, zhaody}@pku.edu.cn

Institute of Computer Science and Technology
Peking University, Beijing 100871, China

Abstract

This paper describes our approaches to real-time filtering task including push notifications on a mobile phone scenario and periodic email digest scenario in the TREC 2015 Microblog track. In the push notifications on a mobile phone scenario, we apply an adaptive timely query-biased filtering framework which utilizes two effective scores to estimate the relevance of tweets. External evidences are well incorporated in our approach with Web-based query expansion technique. In the periodic email digest scenario, we apply pseudo-relevance feedback using language model and similarly we adopt an adaptive dynamic query-biased filtering method to choose the novel representative tweets. Besides, the results of scenario periodic email digest can promote the performance of scenario push notifications since we utilize shared global relevance threshold. Experimental results show that our adaptive query-biased filtering methods achieve good performance with respect to ELG and nCG metrics for push notifications scenario. In addition, our systems for scenario periodic email digest also obtain convincing nDCG scores.

Introduction

Information retrieval in microblogging environment has attracted increasing attention with the growing popularity of microblog. To boost the user experience in the real-time environment, the retrieval results should consist of non-redundant representative tweets during the evolution of a given topic. TREC first introduced Tweet Timeline Generation (TTG) task in 2014 (Lin et al. 2014). The putative user model is given as follows: "I have an information need expressed by a query Q at time t and I would like a summary that captures relevant information.". The system should detect and eliminate redundant tweets and then form a list of non-redundant, chronologically ordered tweets that occur before time t. Follow the idea of TTG, real-time filtering task is proposed in this year's Microblog track to explore technologies for monitoring a stream of social media posts with respect to user's interest profile. The task is composing of two scenarios, namely push notifications on a mobile phone and periodic email digest.

In the scenario of periodic email digest, we apply a language model framework to estimate the relevance between given interest profile with candidate tweets. In the query side, aside from the traditional pseudo relevance feedback based on top ranked tweets, external evidences from the Google search results are also utilized in our retrieval model to better understand the user intent. For each interest profile, we rank the candidate tweets of every day by integrating two retrieval scores which adopt two different smooth methods (i.e. Dirichlet Smooth and JM Smooth) through simple linear combination. We utilize a query-biased adaptive threshold β to choose top K tweets as smaller candidate collection Γ to generate the email digest. For each interest profile Q, the initial threshold β is set as the integrated score of the tenth relevant tweet from the previous day. Meanwhile, we update the threshold of each interest profile everyday with the same way from previous day. For each candidate tweet in collection Γ, we calculate the relevance scores between the candidate tweet with each tweet that has been pushed previously, a tuned novel threshold γ is used to determine whether the candidate tweet is included in the email digest.

In the scenario of push notifications on a mobile phone, different from the other scenario, for each interest profile, when a candidate tweet comes, we immediately determine whether to push the tweet to the interest profile. We first estimate the relevance score between the tweet and interest profile using the normalized KL-divergence distance. For interest profile Q in day D, we utilize the adaptive threshold β in the previous scenario to decide whether the tweet is relevant to Q. Then we compare the tweet with previous pushed tweet in this scenario of Q. Similarly, we use a tuned novel threshold γ to decide whether to push the tweet to the interest profile.

The rest of the paper is structured as follows: We first presents the preliminaries of both scenarios in Section 2. and then we introduce our approaches for periodic email digest scenario in Section 3. In Section 4, we describe our push notifications system in detail. Section 5 present our experimental evaluation. At last, we conclude the paper in Section 6.

Preliminaries

In this section, we first introduce the preliminaries for both scenarios. In the first step, we preprocess the interest profiles and tweet streams.

In the query side, We first obtain external resources for

*Corresponding author.

each user interest profile. Using the Google search API, we obtain the top five items which is consisting of titles and snippets for each interest profile, then we combine the titles and snippets to generate a background context document for each interest profile. In addition, we incorporate the context document with original interest profile through linear interpolation.

In the tweet stream side, our system will monitor the Twitter's live tweet sample stream continuously using the official API. As soon as the system obtains the json data of tweets, the system will preprocess the tweet text and filter non-English tweets or words. To boost the speed of identifying candidate tweets for each user's interest profile, for each interest profile q, we simply filter tweets that do not contain keywords for q, and the rest tweets are chosen as candidate tweet collection Γ for q.

In the following step, we will utilize a query-biased filtering method to choose pushed tweets from the candidate tweet collection Γ for each interest profile every day. In order to push novel tweets for each topic in different days, we maintain a collection consisting of tweets which have been pushed in previous time for each interest profile. Besides, we make a binary classifier to judge the tweets' novelty against the corresponding collection.

Preprocessing

The preprocessing we adopt on the queries and tweet stream is described as follows:

- **Non-English Filtering:** We discard the non-English tweets according to the result of twitter's language detector.
- **Non-English Words:** We simply filter the words which contain non-ASCII characters.
- **Simple Retweet Additional Commentary Elimination:** We eliminate the additional commentary of tweets that contains 'RT @' with the consideration that all retweets should be normalized to the underlying tweets according to the guideline.
- **Porter Stemming and Stopword Filtering:** Stopwords are removed from these tweets using InQuery stopwords list. These tweets are stemmed using the Porter Algorithm.

Query Expansion

As microblog retrieval suffers severely from the vocabulary mismatch problem (i.e. term overlap between query and tweet is relatively small), query expansion techniques can be leveraged to improve the retrieval effectiveness (Zhai and Lafferty 2001). In this section, we introduce the method on the basis of query expansion.

To better describe the query expansion, we first name the original user interest profile (namely query) offered by TREC'2015 *OriginQuery*. Here we only use the topic keywords of each interest profile since we utilize external web resources to depict the background information of the given profile. For a certain *OriginQuery*, we submit the query to Google Search Engine API. Non and verb terms from the titles and snippets of returned top five documents to generate a new query (i.e. *WebQuery*). Then we generate the *MergeQuery* by interpolating the *OriginQuery* and *WebQuery*.

$$MergeQuery = \alpha \cdot OriginQuery + (1-\alpha) \cdot WebQuery \quad (1)$$

Then we utilize the updated query to represent original interest profile and then estimate the relevance between the query and tweets.

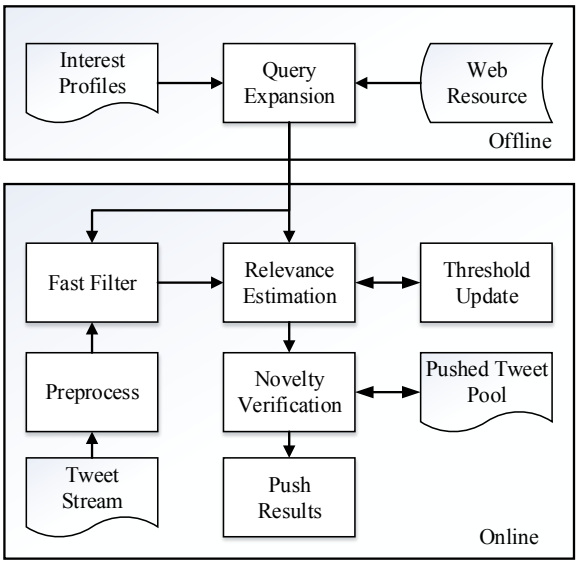

Figure 1: Scenario A System Framework.

Scenario A: push notifications on a mobile phone

System Overview

As noted above, the scenario A push notifications on a mobile phone aims to push relevant and novel tweets to users and such tweets are triggered a relatively short time after the content is generated. In this section, we mainly discuss the architecture of our system, which is shown in 1. From the figure, we can see that our system mainly contains two components:

- **Offline Module:** We utilize the external web resources to obtain context words about each interest profile. Using the Google Web Search API, we obtain top five relevant documents consisting of titles and snippets. Then we preprocess the documents and incorporate with original query to generate a merged query document which has a more comprehensive word distribution.
- **Online Module:** We monitor and preprocess the tweet stream continuously, and we utilize a fast filter module to obtain more possible relevant tweets for each profile. In order to decrease the time delay, as soon as the system gets a possible relevant tweet, the system will immediately estimate the relevance between expanded query

and the tweet. We will update the query-biased relevance threshold every day instead of using a time window. If the system judges a tweet as relevant tweet of a interest profile, the module of Novelty Verification will utilize a greedy clustering algorithm to decide whether the tweet can be regarded a new cluster compared with previous pushed Tweet pool. Once the tweet is regarded as novel tweet, the tweet will be pushed and appended into the pushed Tweet pool.

Fast Filter

In order to boost the speed of identifying possible relevant tweets for each profile, we simply filter tweets that do not contain any keywords for each profile, and the rest tweets are chosen as candidate tweet collection.

Relevance Estimation

We utilize the KL-divergence language model based retrieval method to measure the relevance between query language model $\widehat{\theta}_Q$ and tweet language model $\widehat{\theta}_T$. The smoothing methods we use for language model are: (a) DIR (Bayesian Smoothing with Dirichlet Priors) smoothing, (b) JM (Jelinek-Mercer method) smoothing.

$$LMScore(T,Q) = \sum_{w \in Q} P(w|\widehat{\theta}_Q) \cdot logP(w|\widehat{\theta}_T) \quad (2)$$

We incorporate the two scores using different smooth methods by linear interpolation.

$$\begin{aligned} Score(T,Q) = \lambda \cdot LMScore_{Dir}(T,Q) \\ +(1-\lambda) \cdot LMScore_{JM}(T,Q) \end{aligned} \quad (3)$$

Here we empirically set λ as 0.5.

Adaptive Query-biased Filtering

Considering the fact that different topics can affect different attentation to varying degrees, thus the count of relevant tweets are quite distinct. Thus we utilize two strategies to estimate the query-biased relevance threshold β, namely empirical setting and human assist selection. (1) empirical setting method tries to utilize the popularity and relevance threshold in previous day of each query. Taking advantage of the ranked tweet list in scenario B, in our experiments, we utilize the relevance score of the tweet ranked at top ten as the relevance threshold β in scenario A of next day. (2) Human assist selection also utilizes the ranked tweet list in scenario B, while here human will involve and quickly scan the top 100 tweets (we think top 100 is enough) and decide which one's score is the lower bound. In other words, we will quickly scan the ranked list from top to bottom, once we find one tweet is not relevant, we choose the relevance score of the tweet as the relevance threshold β of the query in the next day.

Since the pushed tweets in the total evaluation period should be non-redundant, we adaptively update the threshold of each interest profile. Here we maintain a pushed tweet pool and utilize a greedy algorithm to determine whether a coming relevant tweet is novel or not (Fei, Hong, and Yang 2015; Albakour, Macdonald, and Ounis 2013). We will calculate the relevance score between coming tweet and all the pushed tweets using the language model described in Relevance Estimation, then we greedily choose the pushed tweet that has highest relevance score with the coming tweet as reference. Once the highest relevance score is less than empirical novel threshold γ, we regard the coming tweet as novel tweet and push the tweet. Finally we append the coming tweet into the pushed tweet pool for estimating novelty of subsequent tweets.

Scenario B: periodic email digest

System Overview

Our approach for this scenario also mainly contains two components:

- **Offline Module:** Similar with Scenario A, we utilize the expanded query to represent the interest profile.
- **Online Module:** We monitor and preprocess the tweet stream continuously, and we utilize a fast filter module to obtain more possible relevant tweets for each profile. Different from scenario A, we maintain a candidate tweet collection for each profile every day. At the end of each day, we will re-rank the candidate tweet collection for each profile which incorporates with pseudo relevance feedback. Once we obtain the ranked tweet list, we have two strategies to determine the query-biased relevance threshold β of next day, thus we can adaptively update the relevance threshold as time goes on. Similarly, we adopt a module of novel verification along with pushed tweet pool to ensure the novelty of recommended tweets as possible.

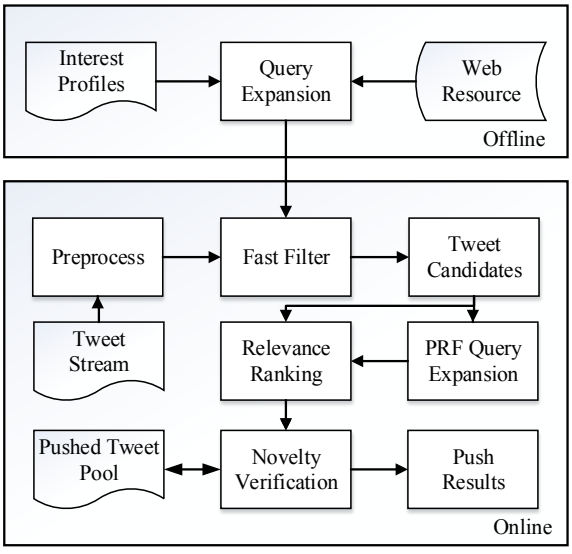

Figure 2: Scenario B System Framework.

The modules of *Fast Filter*, *Relevance Estimation* and *Adaptive Query-biased Filtering* are similar with scenario A.

PRF query expansion

We utilize pseudo relevance feedback to improve the retrieval performance which is widely used in microblog search (Lv and Zhai 2009). Here we directly follow (Liang, Qiang, and Yang 2012) work to incorporate the module in the ranking process.

Result Analysis

Table 1 show the performance of our submitted three runs for scenario A push notifications on a mobile phone. The primary evaluation metric for scenario A is ELG (expected latency-discounted gain) and nCG (normalized cumulative gain) is the second metric. The run PKUICSTRunA1 adopts an adaptive relevance threshold β according to the relevance score at top K in scenario B of previous day. Here we set K as 10. Both PKUICSTRunA1 and PKUICSTRunA2 adopt a empirically uniform novel threshold $\gamma = 0.67$, while PKUICSTRunA2 utilizes an adaptive relevance threshold according to manual selected top position K ($K <= 10$) in scenario B of previous day. Different from PKUICSTRunA2, the run PKUICSTRunA3 uses a empirically uniform novel threshold $\gamma = 0.72$.

From Table 1, we can observe that PKUICSTRunA2 achieves the best performances against ELG and nCG, since the manual relevance threshold and the empirical uniform threshold $\gamma = 0.67$ is effective to depict the needs of non-redundant and relevant tweets. Manual selected top K ($K <= 10$) can increase the precision as it limits the maximum returned tweets of each day, and empirical novel threshold $\gamma = 0.67$ is more suitable that $\gamma = 0.72$.

Table 1: Performance of submitted runs for scenario A

Run ID	ELG	nCG
PKUICSTRunA1	0.1415	0.1566
PKUICSTRunA2	**0.3175**	**0.3127**
PKUICSTRunA3	0.1382	0.1711

Table 2 shows the performance of our submitted three runs for scenario B periodic email digest. The primary evaluation metric is nDCG@K. Among all the three runs, PKUICSTRunB1 utilizes an adaptive relevance threshold β according to the relevance score at position K of the ranked tweet list in scenario B of previous day. Both PKUICSTRunB1 and PKUICSTRunB2 adopt a empirically uniform novel threshold $\gamma = 0.67$, while PKUICSTRunB2 utilizes an adaptive relevance threshold according to manual selected top position K ($K <= 10$) in scenario B of previous day. Different from PKUICSTRunB2, the run PKUICSTRunA3 uses a empirically uniform novel threshold $\gamma = 0.72$.

From Table 2, we can see that PKUICSTRunB3 achieves the slightly better performances than other runs, while all of them do not obtain optimal performances against metric nDCG@K due to the uncertainty of K during the experiments. Since we do not know K during the evaluation period and up to 100 tweets can be returned for each interest profile. However, the returned tweets in a specific day can affect whether some candidate tweets to return due to the redundancy settings. Thus we empirically return relevant and non-redundant tweets according to adaptive novel threshold but we cannot effectively control the returned count of each day. Further investigation and experiments are needed to solve this issue.

Table 2: Performance of submitted runs for scenario A

Run ID	nDCG
PKUICSTRunB1	0.2226
PKUICSTRunB2	0.2228
PKUICSTRunB3	**0.2343**

Conclusion

In this paper, we present our systems for TREC 2015 Microblog track. In the push notification on a mobile phone scenario, we apply an adaptive timely query-biased filtering framework which monitors and estimates the twitter stream with given interest profiles continuously and immediately. In the periodic email digest scenario, We apply pseudo-relevance feedback using language model to rank candidate tweets and then we adopt an adaptive dynamic query-biased filtering method to choose the novel representative tweets every day. Many further investigations and experiments are needed to estimate the sensitivity of relevance threshold and novel threshold, besides, the setting of K is also needed to discuss in the future as soon as we get the ground truth and evaluation scripts.

Acknowledgments

The work reported in this paper is supported by the National Natural Science Foundation of China Grant 61370116.

References

Albakour, M.-D.; Macdonald, C.; and Ounis, I. 2013. On sparsity and drift for effective real-time filtering in microblogs. In *CIKM*, 419–428.

Fei, Y.; Hong, Y.; and Yang, J. 2015. Handling topic drift for topic tracking in microblogs. In *Advances in Information Retrieval*. Springer. 477–488.

Liang, F.; Qiang, R.; and Yang, J. 2012. Exploiting real-time information retrieval in the microblogosphere. In *Proceedings of the 12th ACM/IEEE-CS joint conference on Digital Libraries*, 267–276. ACM.

Lin, J.; Efron, M.; Wang, Y.; and Sherman, G. 2014. Overview of the trec-2014 microblog track. In *Proceedings of the TREC 2014*.

Lv, Y., and Zhai, C. 2009. A comparative study of methods for estimating query language models with pseudo feedback. In *CIKM*, 1895–1898.

Zhai, C., and Lafferty, J. D. 2001. Model-based feedback in the language modeling approach to information retrieval. In *CIKM*, 403–410.