CCNU at TREC 2016 Real-Time Summarization Track

Chao Bei and Po Hu
School of Computer Science, Central China Normal University, Wuhan
430079, China
chaobei@mails.ccnu.edu.cn, phu@mail.ccnu.edu.cn

Abstract

This paper describes our approach to real-time summarization track for push notification scenario and email digest scenario in TREC 2016. This track aims at monitoring a stream of twitter posts and pushing the most relevant tweets to the users according to their interest profiles. In the push notification scenario, we adopt a combined method by take into account several critical factors i.e., relevance, salience and redundancy to select some relevant but non-redundant tweets. In the email digest scenario, in addition to considering these factors, we additionally adopted a novel TF-IDF strategy to automatically rank tweets at the end of a day. The experimental results on both scenarios show the effectiveness of our approach.

1. Introduction

Now there is a large amount of information shared by different social media platforms such as Twitter, Facebook, etc. We have been overwhelmed by big data and it is difficult for us to find useful information effectively and efficiently. Therefore, a system which can automatically monitor the stream of social media posts so that different users with diverse background knowledge may keep up with the latest development of the topics that they care about is of great need. In this case, the feedback information provided by such a kind of system is expected to be relevant, instant and diverse.

This year's track consists of the following two scenarios:

- **Scenario A: Push Notifications.** In this scenario, content that is identified as relevant by a system based on the user's interest profile will be pushed to the mobile phones of users. And the push notifications should be relevant, timely, and novel.
- **Scenario B: Email Digest.** In this scenario, a system will identify a batch of up to 100 ranked tweets per day per interest profile and it is expected that systems will compute the results in a relatively short amount of time after the day ends. Each tweet post which is identified as relevant and novel is to be aggregated into an email digest, which will then be sent to the corresponding user periodically.

In both scenarios, we focus on finding ranked lists of tweets which are both relevant and salient by the classic methods, such as TF-IDF and BM25 [1]. In the push notifications scenario, whenever a tweet comes, we immediately estimate its relevance and redundancy by the computation of JD-divergence, and estimate its salience by a hybrid TF-IDF strategy. In the scenario of email digest, a novel TF-IDF

model [2] is adopted to re-rank the summaries at the end of each day. In addition, we also adopt adaptive thresholds in both scenarios to determine whether a tweet should be pushed or emailed to a specified user.

The rest of paper is organized as follows: we first describe preprocess for both scenarios in Section 2, and then introduce the implementation of our system in detail in Section 3. The parameter settings of our approach is described in Section 4. In Section 5，we present the evaluation performance of our system for both scenarios and analyze the experimental results. Section 6 concludes the whole technical report.

2. Preprocess

In this section, we introduce preprocess of our system for both scenarios. The interest profiles and tweet streams are both preprocessed.

Firstly, we obtain the top ten titles for each interest profile of a user by using Bing search API. Then we combine the titles and the original interest profile to get the top k keywords as the query for each profile. On the handling of twitter stream, our system monitors the twitter's live sample stream continuously by the official API. Once our system gets the tweets, the content of it will be preprocessed.

2.1 Preprocessing

The preprocessing is conducted on both queries and tweet stream. We eliminate all the non-English tweets by a twitter's language detector and the links including a web address is eliminated via regular expressions. Besides, if the number of "#" occurs more than three times, we consider it as a meaningless tweet and eliminate it. All the tweets are tokenized and lowercased. Stop-words are removed through NLTK and other words are stemmed by porter stemmer.

2.2 Query Expansion

Traditionally, query expansion is often used to improve the retrieval effectiveness [3]. In our approach, we first get top five keywords as original keywords from each interest profile. Then, we submit these original keywords to Bing Search API and obtain top k keywords as expansion based on the contents from the top ten titles of retrieval result. The original keywords and expanded keywords are combined to be the new query for each interest profile. Lastly, we use these combined keywords to estimate the relevance between query and tweets.

3. Our System

This section describes the implementation of our system for two scenarios respectively in detail.

3.1 System for Scenario A: Push Notifications

The aim of the push notifications scenario is to push the relevant and novel tweets to user as soon as possible after tweets are published. To achieve this goal, our system contains the following two components:

- **Offline Component:** We get the top ten titles by submitting top five keywords, which we get from each interest profile, to Bing Search API. Getting the top k keywords from these titles as the expanded keywords, we combine the original keywords and expanded ones as the new query for each profile.
- **Online Component:** The system monitors the tweet stream continuously and preprocesses the tweet as soon as the system obtains it. Then by take into account several critical factors (i.e., relevance, salience and redundancy), the system gets the relevant tweets for each interest profile. In order to meet the requirement of users, we update three thresholds of factors in different ways. If a tweet meets the three thresholds, it will be pushed to user and added into the pushed summary A.

3.1.1 Relevance and Redundancy

We use JS-divergence to estimate relevance and redundancy. Relevance estimates the divergence between language model of tweets in stream and language model of queries, while redundancy estimates the divergence between language model of tweets in stream and language model of tweets in summaries. Therefore, the JS-divergence is as follows:

$$\mathrm{W(w)} = \frac{P(w) + Q(w)}{2} \tag{1}$$

$$\mathrm{Relevance(P, Q)} = \frac{1}{2} \sum_{w \in P \cap Q} P(w) \log \frac{P(w)}{W(w)} + \frac{1}{2} \sum_{w \in P \cap Q} Q(w) \log \frac{Q(w)}{W(w)} \tag{2}$$

where P and Q are unigram language models.

And what is different from relevance is that in redundancy there are many summaries pushed, so the redundancy is the minimum of JSD between the tweet in a stream and the tweet in summaries. In our system, the redundancy estimation is as follows:

$$\mathrm{Redundancy}(T_n) = \min_{\forall T_m \in S} Rscore(T_n, T_m) \tag{3}$$

Here we use JM smoothing to avoid zero probability problem in both relevance and redundancy estimation.

3.1.2 Salience

In our system, we compute the salience score based on a hybrid TF-IDF strategy [4]. As describing in [5], we consider that a user may need the novel and salient tweets. So in order to adapt to the large number of tweets and the rapidly rising requirement, the

hybrid TF-IDF is used to estimate salience as follows:

$$\text{Salience}(T_i) = \sum_{w \in T_i} TF(w) \times IDF(w) \tag{4}$$

$$\text{TF(w)} = \frac{\#ofwInAllTweet}{\#WordsInAllTweet} \tag{5}$$

$$IDF(w) = \frac{\#Tweets}{\#Tweets\ wOccurs} \tag{6}$$

3.1.3 Update Threshold

In our system, only the tweet that meets the thresholds of factors can be added into the pushed summaries. And in order to meet the rapidly increasing requirement of user, we update the thresholds in different ways according to different situations. We update the salience threshold δ_A everyday according the salience of the summaries yesterday, and then update the relevance threshold λ_A and the redundancy threshold γ_A when a tweet is pushed to a user.

$$\delta_A^{day+1} = \min_{T_i \in S^{day}} \left(\text{Salience}(T_i)\right) \tag{7}$$

$$\lambda_A = \min_{T_i \in S} Relevance(T_i) \tag{8}$$

$$\gamma_A = \max_{T_i \in S} Redundancy(T_i) \tag{9}$$

3.2 System for Scenario B: Email Digest

The implementation detail of our system in this scenario is similar to that of the scenario A, and it also has two components:

- **Offline Component:** Similarly with scenario A, we use expanded keywords as the new query by means of Bing Search API.
- **Online Component:** Similarly with scenario A, the system monitors the tweet stream continuously and preprocesses the tweet as soon as the system obtains it. Then taking into account several critical factors (i.e., relevance, salience and redundancy), the system gets the relevant tweets for each interest profile. However, how to update the thresholds is different from scenario A. For this scenario, a novel TF-IDF model is used to re-rank the summaries at the end of each day. If a tweet meets the thresholds, we will add it to summary B and email

it to the user periodically.

3.2.1 Update Threshold

Although we still update the salience threshold δ_B in scenario B at the end of each day according the salience of the summaries yesterday and update the relevance threshold λ_B and the redundancy threshold γ_B when a tweet is added into summary B, we adopt different strategy due to the lower requirement compared with scenario A. For this scenario, our strategy of updating thresholds is as follows:

$$\delta_B^{day+1} = \min_{T_i \in S^{day}} \left(\text{Salience}(T_i)\right) \tag{10}$$

$$\lambda_B = \underset{T_i \in S}{avg}\, Relevance(T_i) \tag{11}$$

$$\gamma_B = \underset{T_i \in S}{\text{avg}}\, Redundancy(T_i) \tag{12}$$

3.2.2 A Novel TF-IDF Model

At the end of each day, the relevance between queries and summaries may be changed with the development of the event. In order to estimate relevance better, we use a novel TF-IDF model to re-rank the summaries. The novel TF-IDF model is as follows:

$$\text{RITF}(\text{t}, \text{D}) = \frac{\log_2(1 + TF(t, D))}{\log_2(1 + Avg.TF(t, D))} \tag{13}$$

$$\text{LRTF}(\text{t}, \text{D}) = \text{TF}(\text{t}, \text{D}) \times \log_2(1 + \frac{ADL(C)}{len(D)}) \tag{14}$$

$$\text{BRITF}(\text{t}, \text{D}) = \frac{\text{RITF}(\text{t}, \text{D})}{1 + \text{RITF}(\text{t}, \text{D})} \tag{15}$$

$$\text{BLRTF}(\text{t}, \text{D}) = \frac{\text{LRTF}(\text{t}, \text{D})}{1 + \text{LRTF}(\text{t}, \text{D})} \tag{16}$$

$$\text{w} = \frac{2}{1 + \log_2(1 + |Q|)} \tag{17}$$

$$\text{TTF}(\text{t}, \text{D}) = \text{w} \times \text{BRITF}(\text{t}, \text{D}) + (1 - w) \times \text{BLRTF}(\text{t}, \text{D}) \tag{18}$$

$$\mathrm{IDF}(\mathrm{t},\mathrm{C}) = \ln\left(\frac{CS(C)+1}{DF(t,C)}\right) \tag{19}$$

$$\mathrm{AEF}(\mathrm{t},\mathrm{C}) = \frac{CTF(t,C)}{DF(t,C)} \tag{20}$$

$$\mathrm{TDF}(\mathrm{t},\mathrm{C}) = \mathrm{IDF}(\mathrm{t},\mathrm{C}) \times \frac{\mathrm{AEF}(\mathrm{t},\mathrm{C})}{1+\mathrm{AEF}(\mathrm{t},\mathrm{C})} \tag{21}$$

$$\mathrm{Sim}(\mathrm{Q},\mathrm{D}) = \sum_{i=1}^{|Q|} \mathrm{TTF}(q_i,\mathrm{D}) \times \mathrm{TDF}(q_i,\mathrm{C}) \tag{22}$$

where Avg.TF(t, D), ADL(c), len(D), CS(C) and CTF(t, C) denote the average term frequency of D, the average document length of the collection, the length of the document D, the number of document in collection and the total occurrence of the term t in the entire collection respectively.

4. Our Submissions with Parameter Settings

We submit two runs to compare the effectiveness of our system. In the run1, we use keywords matching to estimate relevance and a tweet containing at least two keywords is considered as a relevant candidate. Therefore, there is no relevance threshold in the run1 and we need more expanded keywords. Then, we set k as 10. On the contrary, JS-divergence is used to estimate relevance in the run2, so we set k as 5. Here, in order to improve the speed of processing, before relevance assessment, we eliminate those tweets that don't contain any keywords.

5. Experimental Results

Different from last year, a new evaluation is added in this year. It is live user-in-the-loop assessments to capture live user assessments. And the traditional post hoc batch evaluation methodology has been refined over the past few years and has been experimental validated. EG0 and nCG0 are added to estimate scores for silent days. Latency and GMP are added to estimate lateness of push notifications and synthesis score of gain and pain respectively. These evaluation is for scenario A, only nCG0 is added into scenario B. EG1 and nCG1 is still the major metrics in the scenario A and scenario B respectively.

Table 1 and Table 2 show live user-in-the-loop assessment and the performance of our two runs on scenario A. Compared with the waterloo baseline, our system performed not so well. The salience assessment of our system is very simple so that it doesn't detect the real salient information and doesn't avoid the information appearing

frequently. But GMP achieves a good performance on all of settings. It means that our system is not good at detecting salient information, but irrelevant tweets are in control because of the strict threshold updating strategies. Besides, there is no significant difference between keywords matching and relevance assessment according the performance for scenario A. They achieve about the same effectiveness.

Table 3 shows the performance of our runs for scenario B. Similarly with scenario A, our system performed not so well. The reason is same with scenario A. Particularly, our run1 is better than run2. It means that keywords matching approach is better than relevance assessment for scenario B.

Table 1. Live user-in-the-loop assessments of our system for scenario A

	CCNU Run1	**CCNU Run2**	**Waterloo Baseline**
#rel	19	17	148
#redundant	0	3	12
#non_rel	95	89	286
#unjudged	728	763	1461
#total_length	842	870	1888

Table 2. Performance of our system for scenario A

	CCNU Run1	**CCNU Run2**	**Waterloo Baseline**
EG1	0.1699	0.1643	0.2289
EG0	0.0003	0.0000	0.0253
nCG1	0.1714	0.1643	0.2330
nCG0	0.0018	0.0000	0.0295
GMP.33	-0.1732	-0.2070	-0.6000
GMP.5	-0.1290	-0.1545	-0.4317
GMP.66	-0.0874	-0.1050	-0.2733

Mean latency	355559.0	0.0	120908.6
Median latency	355559.0	0.0	8718.0

Table 3. Performance of our system for scenario B

	CCNU Run1	**CCNU Run2**	**Waterloo Baseline**
nDCG1	0.1732	0.1554	0.2352
nDCG0	0.0018	0.0000	0.0299

6. Conclusion

In this paper, we present our system for real-time summarization track in TREC 2016. For the scenario A, we expand query via Bing Search API. By taking into account of some factors (i.e. relevance, salience, redundancy), our system determines whether a tweet should be pushed to a user instantly. For the scenario B, we also detect the relevant tweets through the similar factors. But what is different from that of the scenario A is that a novel TF-IDF model is used to re-rank the summaries at the end of each day. Compared with the performance of some baselines, our system performed not so well. Since this is our first time to participate TREC, many further investigations and experiments are needed.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61402191), the Specific Funding for Education Science Research by Self-determined Research Funds of CCNU from the Colleges' Basic Research and Operation of MOE (No. CCNU16JYKX15), and the Thirteen Five-year Research Planning Project of National Language Committee (No. WT135-11).

References

1. Jimmy Lin, Miles Efron, et al. *Overview of the TREC-2015 Microblog Track.* In *Proceedings of The Twenty-fourth Text REtrieval Conference (TREC 2015).* NIST, 2016.

2. Jiaul Hoque Paik. *A Novel TF-IDF Weighting Scheme for Effective Ranking.* In *Proceedings of the 36th international ACM SIGIR,* 343-352.

3. ChengXiang Zhai, John Lafferty. *Model-based Feedback in the Language Modeling Approach to Information Retrieval.* In CIKM, 403-410.

4. Beaux Sharifi, David Inouye and Jugal Kalita. *Summarization of Twitter Microblogs. The computer journal*, 57(3):378-402, 2014.

5. Abdelhamid CHellal, Lamjed Ben Jabeur, et al. *IRIT at TREC Microblog 2015.* In *Proceedings of The Twenty-fourth Text REtrieval Conference (TREC 2015).* NIST, 2016.