ICTIR Subtopic Mining System at NTCIR-9 INTENT Task

Shuai Zhang
Institute of Computing Technology
Chinese Academy of Sciences
Beijing, 100190, P.R.China
zhangshuai01@ict.ac.cn

Kai Lu
Institute of Computing Technology
Chinese Academy of Sciences
Beijing, 100190, P.R.China
lukai@ict.ac.cn

Bin Wang
Institute of Computing Technology
Chinese Academy of Sciences
Beijing, 100190, P.R.China
wangbin@ict.ac.cn

Abstract
This paper describes the approaches and results of our Chinese subtopic mining system for the NTCIR-9 INTENT task. In this system, we first find out the related queries from query logs, then group them into different clusters using a frequent term-set based clustering algorithm. Finally, the central query of each cluster is used to represent the subtopic of this cluster. Encyclopedia and commercial search engines are also used to enhance the mining effectiveness. The evaluation results of our runs show that our approaches perform well. Among the 5 runs we submit, ICTIR-S-C-1 is ranked within top five in terms of D#-nDCG for l=10, 20, 30 and outperforms others in terms of I-rec.

Keywords
subtopic mining, query log, encyclopedia, related searches, frequent term-set, text clustering.

1. Introduction
The NTCIR-9 INTENT task consists of the subtopic mining and document ranking subtasks, aiming to mining different users' various intents based on the given query. In the Subtopic Mining subtask, systems were required to return a ranked list of *subtopic strings* in response to a given query [1].

Based on our experience, users express their intents through queries when using search engines. So our basic idea is that if we can get enough query logs relevant to a specific topic, we can find the subtopics by clustering them. Each cluster represents a subtopic since they have the similar intent. Due to the limitation of the available query logs, several external resources are introduced into our system. It includes online encyclopedia and related searches from search engines. Given a topic, we collect the relevant query logs, the encyclopedia catalogs of the topic and related searches. They are regarded as subtopic candidates and put them into a set of subtopic candidates.

The essential method in our system is one text clustering algorithm. There are a variety of clustering algorithms [2] such as K-means, hierarchal clustering, density-based clustering and so on which can be applied for text clustering. Florian Beil proposed one frequent term-based text clustering method named FTC [2] in 2002. The clustering method we use is similar with FTC since both of them do clustering based on the frequent item sets. Moreover, we utilize different clustering strategies in our implementation which we will discuss later, and the details of FTC can be seen from [2]

Our approach can be described briefly as follows: First of all, subtopic candidates are generated from query logs and out resources. Secondly, cluster them via a frequent term-set based text clustering method. Finally, a ranked list of subtopics can be returned based on the clustering results. The official evaluation results of the NTCIR INTENT show that our system performs well.

The rest of the paper is organized as follows: In the next section, we will introduce the data and external resources we used. Section 3 gives the description of each part of the system in detail. In Section 4, we will present and analyze our official evaluation results. Finally, the conclusion and future work are shown in Section5.

2. Data and resource
What we thought initially was to find the subtopics of a topic from the related query logs. Intuitively, queries entered into search engine contain user intents. However, we realize that some topics have few relevant query logs since the available query logs are limited. To solve this problem, we introduce some external resources from the web. Eventually, we can get enough data for every topic. In brief, the datasets we use are: query logs, catalogs of encyclopedia and related searches from commercial search engines.

The query logs we use include: 1) SogouQ[1] provided by NTCIR INTENT organizers which contains about 30 million clicks collected in June 2008; 2) Sina iask[2] query logs from September, 2006 to October, 2006.

The resources we use can be divided into two categories. The first one is the online encyclopedia. We treat the catalogs of the topic on the encyclopedia webpage as the subtopic candidates. The websites we use are Hudong[3] and Wikipedia[4] (Chinese). E.g., for the topic "*莫扎特" (Mozart)* the subtopic candidates we can get from Hudong are shown in Figure 1:

"莫扎特 简介"
"莫扎特 创作历程"
"莫扎特 人物评价"
"莫扎特 土耳其进行曲"
"莫扎特 小步舞曲"
"莫扎特 单簧管协奏曲"

Figure 1. Example subtopic candidates from Hodong for the topic "Mozart"

[1] http://www.sogou.com/labs/dl/q.html
[2] http://iask.sina.com.cn/
[3] http://www.hudong.com/
[4] http://zh.wikipedia.org/wiki/

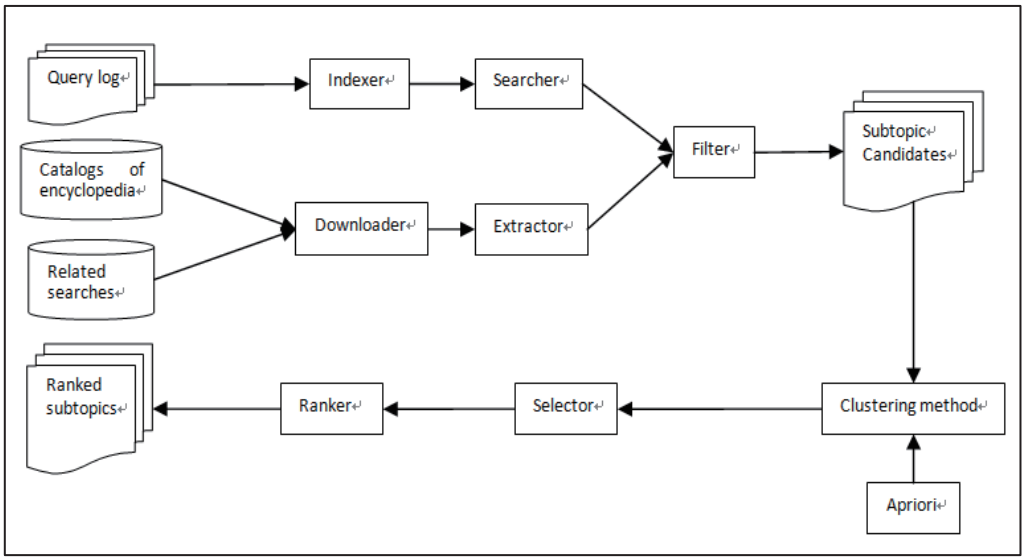

Figure 3. subtopic mining system processing mechanism

The other external resources we use are the related searches from commercial search engines: Baidu[5], Sogou[6] and SoSo[7].Related searches provided by search engines always can reflect users' various intents, so they are very suitable to be subtopic candidates. Related searches can be easily found at the foot of the search result page, as shown in Figure 2.

Figure 2. Example subtopic candidates from Baidu for the topic "Mozart"

3. Subtopic Mining Approach

3.1 System overview

For every topic, we first collect subtopic candidates from query logs, encyclopedia catalogs and related searches from search engines. Then, the frequent term-set based clustering algorithms are conducted to search for subtopics. Finally, each cluster forms one subtopic and one subtopic candidate is selected to represent the subtopic. The processing mechanism of our system is shown in Figure1.

3.2 Data preprocessing

For each topic, the subtopic candidates include the relevant queries from the logs, catalogs of the topic in encyclopedia and related searches from search engines. Preprocessing has a good impact on generating high-quality subtopic candidates.

Query texts in log are indexed by single Chinese character after the same queries are merged. Using single character index can get relevant queries for each topic as many as possible, and it is quite fast at the same time. We employ the Apache Lucene [8] (an open source text search engine library written entirely in Java) to index the query logs. Then, given a topic, we can search for all the relevant queries.

Lucene's default similarity measure is derived from the vector space model (VSM). In the vector space model text is represented by a vector of terms. If $\vec{D}$ is the document vector and $\vec{Q}$ is the query vector, then the similarity between query Q and document D can be represented as:

$$\mathrm{Sim}\left(\vec{D},\vec{Q}\right) = \sum_{t_i \in Q,D} w_{t_{iQ}} w_{t_{iD}} \qquad (1)$$

Where $w_{t_{iQ}}$ is the value of the *i*th component in the query vector $\vec{Q}$ and $w_{t_{iQ}}$ is the *i*th component in the document vector $\vec{Q}$.

For queries that are too long or rare, we consider that they are outliers, which cannot reflect the intent of common users. A heuristic method was introduced to filter the outliers. For each query, we compute the distance between it and the topic, and then filter the queries whose distance is longer than the average. The filtration method is applied only when the total number of relevant queries is larger than the given threshold, MinRelQuery (MinRelQuery = 50 in our system). After this step, some outliers will be filtered. E.g., "红尘逐美酒" and "小窍门红葡萄酒去污法" are removed for the topic "红酒".

The distance measure between two queries we use is the Edit Distance [3]. Given two character strings s1 and s2, the Edit Distance between them is the minimum number of *edit* operations required to transform s1 into s2. The edit operations allowed in our system are:

(1) Insert a character into a string.

(2) Delete a character from a string.

(3) Replace a character of a string by another character.

For the external resources from web, we downloaded the specific html pages and then extracted the content we need. All the work is

[5] http://www.baidu.com/
[6] http://www.sogou.com/
[7] http://www.soso.com/
[8] http://lucene.apache.org/

done automatically and integrated in our system. Some rules are also applied for encyclopedia catalogs to filter the items that are fixed but have nothing to do with the topic, like "参考", "外部链接" and so on.

After the preprocessing step, related query logs, catalogs of encyclopedia and related searches from search engines are collected to form the subtopic candidates set.

3.3 Frequent term-set based clustering

After obtaining subtopic candidates, we cluster them so that candidates whose intents are similar in a cluster. The clustering algorithm we choose is a frequent term-set based method. The idea of this clustering method is quite clear, we find the frequent term sets, and then cluster documents containing the same term sets. Because every term represents a concept and documents in the same clusters include some similar concepts. The method can be described briefly as following:

(1) Segment all the subtopic candidates from text to a set of terms.

(2) Find all the frequent term-sets using Apriori algorithm.

(3) Partition the subtopic candidates set into clusters based on the frequent term-sets.

Frequent items-sets form the basis of association rule mining. In our system, Chinese terms are treated as items, so we will use notion term-set instead of item-set in this paper. We utilize the ICTCLAS library [9](an open source Chinese lexical analyzer) to segment the query string into terms.

In our system, let $T=\{t_1,t_2,\ldots t_m\}$be a term-set and S be the set of all subtopic candidates. Each subtopic candidate is represented by all terms occurring in it. The occurrence frequency f of term-set T is the number of subtopic candidates that contain all the terms in T and the support of T is the percentage. We can compute the support simply by dividing the frequency by the size of S. If the support of a term-set T satisfies a given minimum support threshold (denoted by min_sup) then T is a frequent term-set. If T contains k terms, it's called frequent k-term-set, denoted by L_k.

First of all, we look for all the frequent term-sets via the Apriori algorithm [4]. Apriori employs an iterative approach known as a level-wise search, where k-term-sets are used to explore (k+1)-term-sets. See [4] to know more about the Apriori algorithm, which we won't discuss here.

After the frequent term-sets are found, we partition all the subtopic candidates into clusters based on the frequent term-sets. We tried two partitioning strategies which share the same framework but with small differences. The first method can be described as four steps:

(1) Sort the frequent term-sets descending by k and support.

(2) For each frequent term-sets ts, all the subtopic candidates that contains any terms in ts are selected out to form a cluster. The subtopic candidates selected are removed from the dataset.

(3) Repeat step 2 until no subtopic candidates left or all the frequent terms-set are iterated.

(4) Return the all the clusters.

[9] http://ictclas.org/

The second cluster strategy is different from the first in Step 2.Subtopic candidates that contain all the terms in the frequent term-set are selected to form a cluster instead. That means the first strategy tend to gather bigger cluster for more frequent term-set.

The advantage of this clustering method is that it only has one parameter, min_support, the minimum frequency of a term set. Compared with K-means, it can determine the number of the cluster automatically and we needn't to specify it. As we know, for different topics, their subtopic numbers are also different, so it is very difficult to define the right value of K.

It is the minimum support threshold min_sup, which will affect the number of clusters. Generally, the number of subtopics decreases when min_sup increases. Figure 3 shows the relationship between min_sup and the number of clusters. In the results that we submit, it's based on our experience to specify the value of min_support.

3.4 Subtopic selection and ranking

We can get several clusters of subtopic candidates for every topic. We consider subtopic candidates in a cluster indicating the same subtopic despite that they are a little different in phraseology. Sometimes, some of them may be the subtopic of the subtopic, it's not what we are concern about and wouldn't discuss here.

We choose the central point of the cluster to represent the subtopic. Central point is the point which has the shortest average distance to others in the cluster. The distance measure between two strings is also the Edit Distance that we had mentioned before in Section 2.

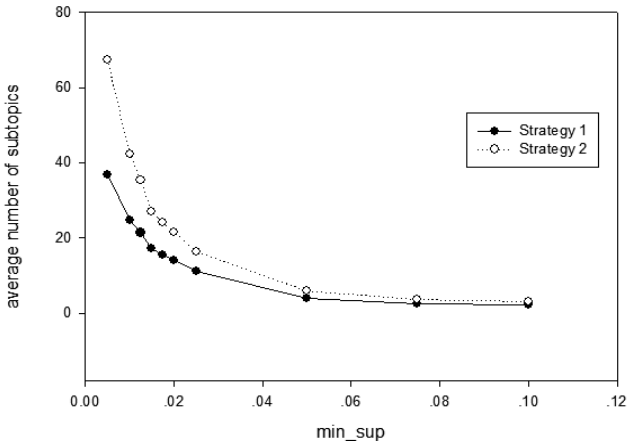

Figure 3. Relationship between min_sup and the average number of subtopics per topic

The last problem left is how to rank the subtopics. In my opinion, it's difficult to say which subtopic is more important than others even for human beings to some extent. So we rank the subtopics by the sizes of their clusters. That means the bigger the cluster is, the more important its subtopic is. Other information such as query times may be used to improve this part in our future work.

4. Results and Analysis

We submitted 5 runs for the subtopic mining subtask. ICTIR-S-C-1, ICTIR-S-C-4 and ICTIR-S-C-5 are the result of the first clustering strategy given in Section 3.3. while ICTIR-S-C-2 and ICTIR-S-C-3 use the second. Moreover, ICTIR-S-C-5 uses query log only while others use external resources.

Table 1. The official subtopic mining results of our system

Run id	I-rec			nDCG			D#nDCG		
	@10	@20	@30	@10	@20	@30	@10	@20	@30
ICTIR-S-C-1	**0.5161(1)**	**0.6997(1)**	**0.7224(1)**	0.6434(14)	0.6162(11)	0.5299(12)	**0.5797(5)**	**0.6579(1)**	0.6261(4)
ICTIR-S-C-2	0.4826(13)	0.6444(3)	0.707(2)	**0.6576(8)**	**0.646(6)**	**0.5895(2)**	0.5701(9)	0.6452(4)	**0.6482(1)**
ICTIR-S-C-3	0.4808(14)	0.5849(18)	0.6062(18)	0.653(11)	0.5913(16)	0.4867(17)	0.5669(12)	0.5881(17)	0.5464(17)
ICTIR-S-C-4	0.5035(3)	0.6206(13)	0.634(15)	0.6417(15)	0.5579(19)	0.4441(21)	0.5726(8)	0.5893(16)	0.539(18)
ICTIR-S-C-5	0.4714(19)	0.5803(19)	0.5924(19)	0.5832(25)	0.5427(22)	0.4407(22)	0.5273(23)	0.5615(21)	0.5165(21)

Table 2. Results comparison to average of all runs and best of other runs

	D#nDCG@10	D#nDCG@20	D#nDCG@30
Average of all runs	0.467352	0.497514	0.473357
Best of the other runs	**0.5993**	0.6519	0.6473
ICTIR-S-C-1	0.5797(24.04%,-3.27%)	**0.6579(32.24%,0.92%)**	0.6261(32.27%,-3.28%)
ICTIR-S-C-2	0.5701(21.99%,-4.87%)	0.6452(29.68%,-1.03%)	**0.6482(36.94%,0.14%)**

Table 3. The average number of subtopics for 4 runs results

Run id	Average number	min_sup	Strategy
ICTIR-S-C-1	36.89	0.005	1
ICTIR-S-C-2	42.90	0.01	2
ICTIR-S-C-3	17.14	0.02	2
ICTIR-S-C-4	21.60	0.015	1

Table 1 shows the evaluation results for all runs, respectively for three metrics: D#nDCG [6], intent recall (I-rec) and nDCG. D#nDCG is the primary metric. In Table 1, the best results are highlighted and the numbers encapsulated in parentheses are the ranks among totally 48 runs in Chinese subtopic mining task.

As is shown in Table 1，ICTIR-S-C-1 and ICTIR-S-C-2 are better than other runs in our system. We also compare the two results with the average values of the runs of all the teams and the best results of other teams in Table 2. The values in the parentheses are the improvement percentage on the average results and other teams' best results. As Table 2 shows, the improvement on the average results range from 21.9% to 36.9%. Moreover, ICTIR-S-C-1and ICTIR-S-C-2 achieve the best results among all the teams when l = 20, 30 respectively.

In general, our system shows good performance. As the results show, the system performs better when l=20, 30 than that when l=10. That means our rank algorithm is not so effective as to rank the more relevant subtopics to top 10.

After the analysis of I-rec and nDCG separately, we can find that our system outperforms others with respect to the evaluation metric of I-rec. ICTIR-S-C-1 achieves the highest I-rec score for l =10, 20, 30 among all runs. But the results for nDCG are not so good. The nDCG scores of ICTIR-S-C-2 rank 8th, 6th, 2nd respectively for l=10, 20, 30, which are the best among the five runs we submit. So we can draw a conclusion that the overall relevance across intents of our system needs to be improved.

On the other hand, we submitted 2 runs for each cluster strategy. ICTIR-S-C-1 is better than ICTIR-S-C-4 while ICTIR-S-C-2 is better than ICTIR-S-C-3. Table 4 shows the average number of subtopics acquired of each run. We can see that ICTIR-S-C-1 and ICTIR-S-C-2 have returned more subtopics. According to the results, it seems that more subtopic returned better score will achieve.

Finally, ICTIR-S-C-5 are the subtopic-mining results by using query logs only without any other external resources. Obviously, it works worse than the other 4 runs, indicating that the external resources bring an improvement for the effectiveness of the system.

5. Conclusion and Future work

In this paper, we describe our system for INTENT task in NTCIR-9. We focus on the Chinese subtopic mining subtask.

In our system, firstly, we acquire subtopic candidates from query logs, encyclopedia and related searches from search engines. Then, we cluster the subtopic candidates by a method based on frequent term-set. Next, one subtopic is selected from each cluster to represent the subtopic. Finally, we rank the subtopics by the size of the clusters which they belong to and return the list.

Our system achieves good performance in the official evaluation, especially on the intent recall metric. However we need to improve the overall relevance across intents. In the future, we will improve the subtopic ranking algorithm and introduce more features to help ranking subtopics. We will also try other clustering algorithms to compare their results, and use ensemble learning methods to combine together which might be more effective. Some other string similarity measures such as the longest common subsequences will be tried and compared to make the system better.

6. ACKNOWLEDGMENTS

The authors thank the NTCIR committee to hold the NTCIR-9 INTENT task. Thank all the members in the Information Retrieval Group of Institute of Computing Technology, Chinese Academy of Sciences for their advice and support. Thank Mr. Zhewei Mai for his effort in proofreading our manuscripts.

7. REFERENCES

[1] Ruihua Song, Min Zhang, Tetsuya Sakai, Makoto P. Kato. Overview of the NTCIR-9 INTENT Task, to appear,2011

[2] P. Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA, 2002

[3] Florian Beil , Martin Ester , Xiaowei Xu, Frequent term-based text clustering, Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, July 23-26, 2002, Edmonton, Alberta, Canada

[4] Walter F. Tichy, The string-to-string correction problem with block moves, ACM Transactions on Computer Systems (TOCS), v.2 n.4, p.309-321, Nov. 1984

[5] Jiawei Han , Micheline Kamber, Data mining: concepts and techniques, Morgan Kaufmann Publishers Inc., San Francisco, CA, 2000

[6] T. Sakai and R. Song. Evaluating Diversified Search Results Using Per-Intent Graded Relevance. In Proceedings of ACM SIGIR 2011, pages 1043–1052, 2011.