FRDC at the NTCIR-11 IMine Task

Zhongguang Zheng
FRDC, China
zhengzhg@cn.fujitsu.com

Shuangyong Song
FRDC, China
shuangyong.song@cn.fujitsu.com

Yao Meng
FRDC, China
mengyao@cn.fujitsu.com

Jun Sun
FRDC, China
sunjun@cn.fujitsu.com

ABSTRACT

The FRDC team participated in the IMine task of the NTCIR-11, including subtopic mining and document ranking subtasks for Chinese language. In the subtopic mining subtask, we propose two methods to build the two-level hierarchy subtopics. Our methods gain high *F-score* and *H-score* respectively. In the document ranking subtask, we adopt various features for relevant webpage retrieval and document ranking.

Team Name

FRDC

Subtasks

IMine(Chinese)

Keywords

FRDC, subtopic mining, document ranking

1. INTRODUCTION

The FRDC (Fujitsu Research and Development Center) team participated in the both subtasks of the IMine task in the NTCIR-11, including subtopic mining and document ranking [5] for Chinese language.

In the subtopic mining subtask, our goal is to build a two-level hierarchy of underlying subtopics for the given queries. The queries could be ambiguous, broad or clear. In our experiment we handle all kinds of queries in the same way. We adopt mainly two strategies for the subtask. One is merely based on the document clustering technology and uses a novel method to refine the document clustering result. This strategy is effective to find first-level subtopics of ambiguous queries (high *F-score*), but fail to gain good performance in *S-score* and *H-score*. The other applies BaiduPedia as an external knowledge source so that classification method can be used for subtopic disambiguation. This strategy gains high *H-score*.

In the document ranking subtask, our goal is to find a ranked list of webpages based on the subtopic mining result. We exploit various features to measure the relevance of the webpage to the subtopic, such as the coverage of the subtopic key words in the webpage. A manually labeled classifier is applied to decide whether a webpage is related to a certain subtopic. Other features are adopted for the webpage ranking, such as the feature list of Microsoft Research[1].

Table 1: Date set for subtopic mining.

Resource	Official	Size	Training Document
Test Set	yes	50	–
QuerySuggestion	yes	1853	Google
RelatedQueries	yes	2321	Google
BaiduPedia Entries	no	117	BaiduPedia

2. SUBTOPIC MINING

2.1 Data Set

Table 1 lists all the data sets for the subtopic mining subtask. Column *Resource* is the resource name. *Official* means whether the data set is provided by the official organization. *BaiduPedia Entries* are the entries of the test set queries from BaiduPedia[2]. *Size* is the number of queries. As to *Training Document*, *Google* means using top 10 Google search result as the training document for each query. *BaiduPedia* is the entry page of the query from BaiduPedia.

We adopt two different strategies for subtopic mining subtask. The first is based on document clustering technology and there is no external knowledge involved. The second exploits BaiduPedia as the knowledge base and uses both document clustering and classification technologies.

2.2 Document Clustering (DC) Method

Firstly, we cluster the candidate queries to get the second level subtopics, and then generate the first level subtopics basing on the second level results. This method contains following steps.

Step 1. Convert training document into word vector. After word segmentation, the document can be changed into word vector presentation. TF-IDF is adopted as the weight scheme. In this way, the one query is presented by the word vector from its training document.

Step 2. Initial clustering. For all the queries in Table 1 (except for the test set), we firstly use document clustering to generate the second level subtopics. An open source toolkit *Cluto*[3] is adopted. In order to find the optimal k, we set k from 2 to 10 and then select the best result through the following methods.

[1] http://research.microsoft.com/en-us/projects/mslr/feature.aspx

[2] http://baike.baidu.com/

[3] http://glaros.dtc.umn.edu/gkhome/views/cluto

Step 3. Refine clustering result. Suppose that we have clustered queries into k classes c_1,c_2,...,c_k. Then we run LDA [1] model to obtain the topic words of c_r. The topic number is set by 2. Thus we will get topic word lists t_0 and t_1. If c_r is well formed, t_0 and t_1 should be similar, and vice versa. We calculate the cosine similarity s between t_0 and t_1 and will split one cluster into two if its s score is lower than a threshold. Now the clustering result may be changed into c_1,c_2,...,c_m, then again, we run LDA model to obtain topic word list for c_r ($1 \leq r \leq m$). After splitting, some similar clusters may be generated. We will merge c_i and c_j if the cosine similarity $sim_{topiclist}(t_i, t_j)$ between their topic word list t_i and t_j is higher than a threshold. The mergence is an iterative process till there is no new cluster generated. In our experiment LDA model is run by *GibbsLDA++*[4] toolkit.

Step 4. Select optimal clustering result. The optimal clustering result is decided by an inner distance $dist_{inner}$ score described as:

$$dist_{inner} = \frac{\sum_{d_i,d_j \in c_r} sim_{topiclist}(t_i, t_j)}{|c_r|} \tag{1}$$

where d_r is one document in c_r. $sim_{topiclist}(t_i, t_j)$ is calculated in the prior step and $|c_r|$ is the document number of c_r.

Step 5. Sort second level subtopics. Suppose c_r is the optimal clustering result. t_c is the topic word list of c_r obtained by LDA model. d_i is the ith document in c_r. wv_i is the word vector presentation of d_i. The second level subtopics are sorted according to the cosine similarity between wv_i and t_c.

Step 6. Generate first level subtopic. Suppose that query "先知*(prophet) dota*" and "先知*(prophet)* 出装*(equipment)*" belong to the same cluster and they are all about computer game, thus the key phrase "*computer game*" should be mentioned frequently in their corresponding training documents. Thus we adopt some unsupervised technologies to extract meaningful phrases from the training documents as the first level subtopics. Inspired by [2][3] and [4], *Accessor Variety* and *C-value* are adopted. The AV value is defined as:

$$AV(s) = min\{L_{av}(s), R_{av}(s)\} \tag{2}$$

where $L_{av}(s)$ and $R_{av}(s)$ denote the number of the distinct predecessor and successor words of phrase s. The *C-value* algorithm is described as:

$$C - value = \begin{cases} log_2|a| \cdot f(a) & \\ & a \ is \ not \ nested, \\ log_2|a|(f(a) - \frac{1}{P(T_a)}\sum_{b \in T_a} f(b)) & \\ & otherwise \end{cases} \tag{3}$$

where a is the candidate phrase, $f(.)$ is the frequency of a. T_a denotes the phrase set that contains a as substring. $P(T_a)$ is the size of T_a. We select the phrase with the highest AV and *C-value* scores. In this way, we obtain the two-level hierarchy subtopics.

2.3 Classification & Clustering (CC) Method

BaiduPedia is exploited as an external knowledge base in this method. We consider BaiduPedia entries of the test set as the first level subtopics. This method includes the following steps.

Step 1. Vectorize the training documents. This step is almost the same as the DC method. The only difference is using LDA model to obtain topic words as the word vector.

Step 2. Classification. BaiduPedia entries are considered as the first level subtopics and the entry pages are exploited as the knowledge base. We just classify all the candidate queries according to the knowledge base. The classification process is based on the rules below.

- For a topic with more than 1 Baidupedia entries, we take each Baidupedia entry as one class, and 1NN method with Euclidean distance is used to classify the second level candidate subtopics about this topic.
- For a topic with just 1 Baidupedia entry or with no Baidupedia entry, we take all the candidates as one class.

Step 3. Document clustering. A threshold-based clustering method is designed to cluster candidate second level subtopics. We randomly sample some "*candidate couple*", and the threshold in the clustering method is calculated as the average value of the Euclidean distance between two candidates in each "*candidate couple*". The sampling number is defined as below:

$$s_n = \lceil \frac{c_n(c_n - 1)}{2}/100 \rceil \tag{4}$$

in which the c_n means the number of candidate second level subtopics, and accordingly $\frac{c_n(c_n-1)}{2}$ means the number of all possible "*candidate couple*". We get the one percent of this number, and ceil it to be an integer as s_n. After getting the threshold, we take all candidates which have smaller similarity than the threshold with each other as one class.

Step 4. Merge the classification and the clustering results. For each cluster, we put the word series of the query items in this cluster together, and detect a most similar BaiduPedia entry with the smallest Euclidean distance. Then we judge the relation between this cluster and the BaiduPedia item with rules below:

- If half or more than half query items in this cluster belong to this BaiduPedia in the classification result, we judge that this cluster is related to this BaiduPedia entry, and take this BaiduPedia entry as the first level subtopic of this cluster.
- If less than half query items in this cluster belong to this BaiduPedia in the classification result, we judge that this cluster is not related to this BaiduPedia entry, and we extract frequent keywords as the first level subtopic name of this cluster.

Based on the above rules, we finally take each cluster as a group of second level subtopics, and either a BaiduPedia entry or a keyword will be taken as the name of the corresponding first-level subtopic.

Step 5. Subtopic ranking. We first calculate the *ranking score* of the second level subtopics. Since all the second level subtopics are real queries, we can easily get the number of web search engine results for each subtopic, and this is the only factor for ranking. For score normalizing, we empirically design the below formula:

[4] http://gibbslda.sourceforge.net/

$$rs = \frac{log(pn_{st}+1)}{log(max\_pn_{st}+1)} \quad (5)$$

where the pn_{st} means the webpage number of web search engine results for a second level subtopic, and the max_pn_{st} means the maximum pn_{st}, and the rs will be a real quantity between 0 and 1.

Then we calculate the weight value of a first level subtopic with the sum of its related second level subtopics' weight value, and process a normalizing step for the first-level subtopics.

According to the weight value based ranking result of first level subtopics and second level subtopics, we keep at most 5 first-level subtopics for each topic, and at most 10 second level subtopics for each first level subtopic as the final submission result.

2.4 Experimental Results

Our team submitted 5 groups of results. The description of the results are listed below:

- *FRDC-S-C-1A*. Result of *DC* method.
- *FRDC-S-C-2A*. Result of *CC* method.
- *FRDC-S-C-3A*. Almost the same as *FRDC-S-C-1A*. The difference is the selection of second level subtopics. When observing the result, we find that some second level subtopics are similar in meaning with each other and they are all highly ranked. Since only at most 10 second level subtopics are allowed, some different subtopics may be ignored. In order to obtain broader subtopics, we take a further step basing on *FRDC-S-C-1A*. We re-cluster the second level subtopics into three classes and sort them according to the class size. Then we select subtopics from the three classes according to the quantity 5, 3 and 2.
- *FRDC-S-C-4A*. Result of section 2.3. Considering the low quality of RelatedQueries, we just use the QuerySuggestion as the second level candidate subtopics with a merging of similar queries, and all other steps are same with *FRDC-S-C-2A*.
- *FRDC-S-C-5A*. We undo the merging of similar queries, and all other steps are the same with *FRDC-S-C-4A*.

The official evaluation method [5] is as follows:

$$H-measure = Hscore*(\alpha*Fscore+\beta*Sscore) \quad (6)$$

The official evaluation of our results are listed in Table 2 and Table 3. Column *Runs* denotes the submitted results with the same prefix *FRDC-S-C*. *Method* is the main scheme of each result. From the results, we can observe that, the *DC* method is effective to solve ambiguous queries. *FRDC-S-C-1A* and *FRDC-S-C-3A* achieve the best *F-score* among all the results and *F-score* is only evaluated on the ambiguous queries. This means our method could effectively generate the first level subtopics for ambiguous queries.

However, the *DC* method does not work well on broad/clear queries. Through the comparison with the official SM result, we find that our method produce much larger granularity results. Taking query "野葛根*(kudzuvine root)*" for example, we generate the first level subtopics including brand "汤臣倍健*(By-Health Co., Ltd.)*", origin "泰国*(Thailand)*" and product "胶囊*(cell)*". Reference topic "功效*(effect)*" is missing because it is already contained in the topic "泰国*(Thailand)*". The training documents talking about the origin are likely to mention the effect at the same time.

Table 2: Official results of subtopic mining sorted by H-Measure.

Runs	H-Score	F-Score	S-Score	H-Measure	Method
5A	0.5377	0.5004	0.3139	0.1757	CC
4A	0.5436	0.4782	0.2715	0.1724	CC
1A	0.2931	0.7191	0.3110	0.1327	DC
3A	0.2897	0.7191	0.3214	0.1326	DC
2A	0.3257	0.5045	0.2381	0.1032	CC

Table 3: Ranking list of subtopic mining sorted by H-Measure.

Runs	H-score	F-Score	S-Score	H-measure	Method
5A	5/19	11/19	12/19	8/19	CC
4A	**3/19**	14/19	14/19	11/19	CC
1A	17/19	**1/19**	13/19	15/19	DC
3A	18/19	**1/19**	11/19	16/19	DC
2A	16/19	9/19	15/19	19/19	CC

As a result, the clustering method gets the two topics together. This results in low *F-score* for *FRDC-S-C-1A* and *FRDC-S-C-3A*. The broad/clear queries should be treated in other ways.

FRDC-S-C-4A based on the *CC* method gains high *H-score*. The BaiduPedia provides solid prior knowledge for the both ambiguous and broad/clear queries. As a result, *FRDC-S-C-4A* has a better performance on *H-score*.

However, both methods fail to yield satisfactory results on *S-score*. Our ranking methods are too naive. Such as in *DC* method, we only consider the content similarity between a query and the cluster it belongs to. More other information should be applied.

3. DOCUMENT RANKING

3.1 Data Set

The only data we use is the SogouT(Ver. 2008) provided by the official organization. The input for this task contains all the five outputs of SM subtask.

3.2 Methodology

In the document ranking task, we exploit various features for relevant page retrieval and ranking. Our method is described below.

Step 1. Data preparation. We extract three parts from the webpage:

- Title: It is known that the title of the webpage contains important information.
- Anchor: The information between the "*meta*" tag in

the HTML webpage. It may contain topic/domain information of the webpage.

- Body: The text content of the webpage.

Step 2. Query expansion. When we input a query, we want to add more information to make the retrieval result more accurate. Thus we want to extract some key words that are related to the given query. For example, when retrieving query "先知(prophet) dota", we want to input more key words such as "computer game" so as to find more relevant webpage. We extract key words in several ways:

- Run LDA model on the training documents of the candidate queries to obtain the topic word list as key words.
- Extract high frequency segmentations from training documents as key words and eliminates single character segmentations.
- Extract segmentations with high TF-IDF value as key words. Combine single character segmentations with other key words generated together to build new key words.
- According to the two-level hierarchy structure of SM result, for the query below a subtopic that keywords has been extracted, add its keywords to the subtopic.

Step 3. Feature selection for document ranking. We exploit various features for ranking the webpage. The features are described below:

- Query coverage. The segmentation number of a query that one webpage covers in the title/anchor/body parts.
- Keywords coverage. This value is calculated in the title/anchor parts of one webpage. The formula is:

$$KeyCoverage = \frac{\sum_{i=0}^{N} K(w, key_i)}{|key| log(|\overline{w}|)} \qquad (7)$$

  w is the title of the webpage, $K(x, y)$ equals to 1 only when keyword i exists in w, operator $|\overline{w}|$ calculates the length of the string except keywords. When the number of keyword in the string less than two, the value is zero.

- TF-IDF similarity. Calculate the cosine similarity between the body of candidate webpage and the training document of the query.
- Keywords weight of the body part in a webpage.

$$KeywordWeight = \sum_{i=0}^{N} \sum \{C(w, key_i) * TFIDF(key_i) * \alpha(t)\} \qquad (8)$$

  key_i is the ith keyword. $TFIDF(x)$ is the TF-IDF value of x. $C(x, y)$ represents searching y in un-segmented sentence x recursively and accumulating the count. First match is 1, the second is 2 and this process goes on. $\alpha(t)$ is a depress factor smaller than 1 and decreasing with the times that y appear in x.

Table 4: Official results of document ranking.

Runs	Coarse-grain results	Fine-grain results	Method
1A	0.4619	0.4118	DC
3A	0.4440	0.3950	DC
2A	0.3899	0.3402	CC
5A	0.3841	0.3338	CC
4A	0.3746	0.3240	CC

Table 5: Ranking list of document ranking.

Runs	Coarse-grain results	Fine-grain results	Method
1A	4/10	4/10	DC
3A	5/10	5/10	DC
2A	6/10	6/10	CC
5A	7/10	7/10	CC
4A	8/10	8/10	CC

- Features inspired by Microsoft Research. We calculate the sum of term frequency, min of term frequency, max of term frequency, mean of term frequency and variance of term frequency.

  Step 4. Webpage classification. In order to judge whether a webpage is related to a query or not, we manually labeled 100 query-document pairs for training a classifier. The output of the classifier are *not related*, *marginally related* and *related*. We use the SVM classifier in our experiment.

  Step 5. Document ranking. We calculate and normalized the above features, then the sorted result is generated.

The official document ranking result is listed in Table 4 and Table 5. We submitted five groups of results based on each subtopic minging result.

4. CONCLUSIONS

In the subtopic mining task, our two methods achieve high *F-score* and *H-score* respectively. However, our topic ranking method is not yet mature. In the document ranking task, we exploit various features for relevant webpage retrieval and document ranking.

5. REFERENCES

[1] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet Allocation. Journal of Machine Learning Research, pages 993-1022. 2003.

[2] Frantzi K, Ananiadou S. Extracting Nested Collocations. in Proc. of the 16th Conference on Computational Linguistics, pages 41-46. 1996.

[3] K. Frantziy, S. Ananiadou, and H. Mimaz. Automatic Recognition of Multi-Word Terms: the C-value/NC-value Method. International Journal on Digital Libraries, pages 117-132. 2000.

[4] Zhao, H., Kit, C. Unsupervised segmentation helps supervised learning of character tagging for word segmentation and named entity recognition. In the

Sixth SIGHAN Workshop on Chinese Language Processing, 2008.

[5] Yiqun Liu, Ruihua Song, Min Zhang, Zhicheng Dou, Takehiro Yamamoto, Makoto Kato, Hiroaki Ohshima and Ke Zhou. Overview of the NTCIR-11 IMine Task. 2014.