THUIR at NTCIR-12 IMine Task

Zeyang Liu, Ye Chen, Rongjie Cai, Jiaxin Mao, Chao Wang, Cheng Luo, Xin Li,
Yiqun Liu*, Min Zhang, Huanbo Luan, Shaoping Ma
Tsinghua National Laboratory for Information Science and Technology,
Department of Computer Science & Technology, Tsinghua University, Beijing, China
yiqunliu@tsinghua.edu.cn

ABSTRACT

In this paper, we describe our approaches in the NTCIR-12 IMine task, including Chinese Query Understanding and Chinese Vertical Incorporating. In Query Understanding subtask, we propose different strategies to mine subtopic candidates from a wide range of resources and present a two-step method to predict the vertical intent for each subtopic. In Vertical Incorporating subtask, we adopt a probabilistic algorithm to rerank the result lists of Web documents and incorporate virtual verticals into the result lists based on the intent of subtopic behind the query.

Team Name

THUIR

Subtasks

Query Understanding(Chinese), Vertical Incorporating(Chinese)

Keywords

query intent, subtopic mining, vertical incorporating, document ranking

1. INTRODUCTION

In NTCIR-12, THUIR group participated in IMine task, which includes query understanding and vertical incorporating [5] for Chinese language.

In Chinese Query Understanding subtask, the goal of our task is to generate a diversified ranked list of subtopics with corresponding vertical intents. In the first step of our experiment, we try to adopt different strategies to mine subtopic candidates (see Figure 1). Similar to the approach in [2], we extract subtopic candidates from Wikipedia disambiguation items, query facets, query reformulations, query recommendations and some other resources. We adopt K-means algorithm to cluster the candidates from different resources into several clusters and use learning-to-rank algorithm to rank the subtopic candidates to get the ones of high quality. After the subtopic lists are generated for each query, we propose a two-step framework to predict the vertical intents of these subtopics: (1) We extract some candidate queries with vertical intents from a Web directory and generate a training dataset based on the original search result pages (SERPs) of these candidate queries. A logistic regression algorithm is adopted to predict the potential vertical intents for each subtopic. (2) According to the distribution of vertical results on SERPs, we further present a rule-based method to diversify the vertical intents of subtopics.

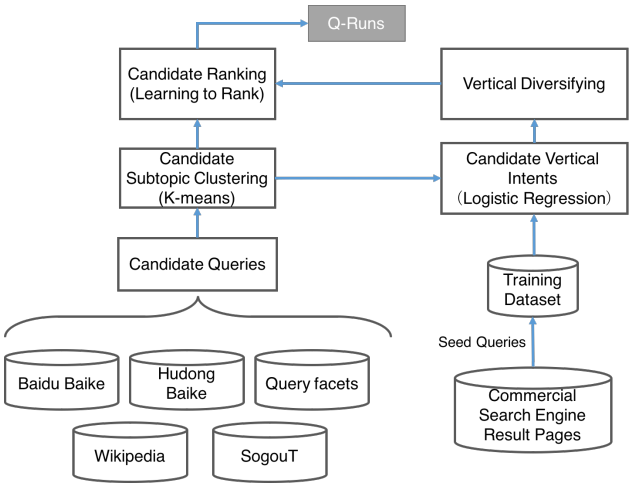

Figure 1: Framework overview for Chinese Query Understanding

In Vertical Incorporating subtask, we first implement a probabilistic model to rerank Web results based on the subtopic mining results in Query Understanding subtask. Considering the virtual vertical is relevant to only one subtopic, we construct a model to describe the connection between the verticals and the subtopics. Then we calculate the importance of verticals for each query and incorporate these virtual verticals into the ranked lists of Web documents.

2. CHINESE QUERY UNDERSTANDING

2.1 Candidates Mining

There are many information resources on the Internet from which we can extract both subtopics and their popularity of an ambiguous query. The information we used in this work include:

- Baidu Baike[1] disambiguous items
- Hudong Baike[2] disambiguous items

*Corresponding author

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

- query facets provided by organizers, including query completions and query suggestions
- Wikipedia disambiguous items
- query reformulations extracted from SogouT dataset[3]
- query recommendations from Sogou [4]

To extract query reformulations for query subtopic, we first use the method described in [1] to segment query logs in SogouT dataset into search sessions and then select the sessions which contains the task queries. Then the queries in the search sessions will be used as the query reformulations of the corresponding query topic.

All generated candidates are cleaned by a sequence of filters to select the high-quality candidates. Each filter focuses on a specific aspect of the candidates. For example, the first filter will remove all the stopwords from candidates to calculate a more appropriate semantic similarity. Candidates longer than 20 characters are also removed by a filter because such candidates may be too verbose to represent a subtopic of the original query.

2.2 Candidate Subtopic Clustering

We implement a K-means model based on scikit-learn[5] to cluster candidates into several subtopic classes because there are many duplicates among candidates mined from different resources which may carry similar query intents. We use the word2vec toolkit[6] to train word embeddings based on the SogouT dataset. For long candidate which is composed of several words, we use the average word embedding of those words as its word embedding representation.

Then a K-means algorithm is applied to cluster these subtopic candidate vectors into n (set as 5 or 10 in our work) clusters $c_1, c_2, ..., c_n$. We try to adopt some metrics to evaluate the quality of each subtopic candidate. We first calculate an inner distance $S_{inner}(q)$ for each candidate q (which belongs to cluster c_i) as shown in Equation 1 and an outer distance $S_{outer}(q)$ as shown in Equation 2 to measure the quality of each cluster.

$$S_{inner}(q) = \frac{\sum_{q_k \in c_i, q_k \neq q} dist(q, q_k)}{|c_i| - 1} \quad (1)$$

$$S_{outer}(q) = min\{\frac{\sum_{q_k \in c_j} dist(q, q_k)}{|c_j|}\}(j = 1, 2, ..., n, j \neq i) \quad (2)$$

$$S(q) = \frac{S_{outer}(q) - S_{inner}(q)}{max\{S_{outer}(q), S_{inner}(q)\}} \quad (3)$$

Based on S_{inner} and S_{outer} we can define a score $S(q)$for each candidate q to measure its importance to the original search query. As shown in Equation 3, the larger $S_{outer\_j}(q)$ is, and the smaller $S_{inner}(q)$ is, the higher $S(q)$ will be. Intuitively, we consider a subtopic candidate better if it is more different from those candidates from other clusters and more similar with those candidates in the same cluster. It is reasonable to regard the candidates with higher $S(q)$ as better ones.

2.3 Candidate Ranking with Learning to Rank

Considering that the clustered candidates generated by K-means algorithm may contain some noises, we further adopt learning to rank algorithm to rank the subtopic candidates to reserve the ones of high quality. We use the evaluation results of NTCIR-11 IMine task to be our training set. In the NTCIR-11 IMine evaluation results, there are 641 second level subtopics available and the subtopics are sorted according to their relevance to the query. So we apply the subtopics and their rankings to train the learning-to-rank model. The features we adopted to train the model can be grouped into three categories:

- **Text similarity-based**: edit distance, jaccard distance and length difference between the query and the candidate.
- **Word embedding-based**: Cosine similarities between the query and candidate embeddings. Fine-grained cosine similarities of word embeddings are also used to train the model, which means if the query is composed of m words after segmentation[7] and the candidate is composed of n words after segmentation, then we can get $m \times n$ cosine similarities. We use the average, medium, top 3 average and top 5 average of these $m \times n$ cosine similarities as the features to train the model.
- **Search result-based**: Number of shared search results of the query and subtopic candidate.

Considering we need to submit 10 query intents for each query, so we choose NDCG@10 to be the optimization metric. We adopt 5-fold cross validation and compare the performance of different learning-to-rank algorithms and normalization methods. The results are shown in Table 1.

From the table we can see that RankBoost algorithm without normalization performs the best, so we adopt this algorithm to train the model based on NTCIR-11 IMine dataset and use it to rank the candidates in our task. Please be noted that we just adopt the learning to rank algorithm in some of our submitted runs and in these runs, we use the relevance score generated by learning to rank algorithm to replace $S(q)$ defined in Equation 3 for later use.

2.4 Vertical Predicting

To identify the relevant verticals for each subtopic, we adopt a two-step method to predict vertical presentation, which includes the vertical predicting step and the vertical diversifying step. In the vertical predicting step, we construct prediction models based on a Web directory and search result pages, and select candidate verticals according to the *presentation score* for each subtopic. However, because Web presentation may also contain vertical intents in some cases, this step is not enough to cover sufficient vertical intents. For example, subtopic "iPhone 6 review" which is presented in Web presentation (given by official guideline) may also contain news intent and QA intent. To reduce this strong effect of Web presentation bias, we further diversify

[3]http://www.sogou.com/labs/dl/t-e.html
[4]A famous commercial search engine in China
[5]https://github.com/scikit-learn/scikit-learn
[6]https://code.google.com/archive/p/word2vec/
[7]The segmentation tool used in our work is jieba, https://github.com/fxsjy/jieba

Table 1: Learning-to-rank results of candidate ranking in QU subtask

Method	No Normalization	Normalized by sum	Normalized by z-score
MART	0.5133	0.5426	0.5043
RankNet	0.5365	0.5328	0.6221
RankBoost	**0.6618**	0.6560	0.5467
AdaRank	0.5428	0.5560	0.5468
Coordinate Ascent	0.6157	0.5839	0.6352
LambdaRank	0.5349	0.4943	0.5796
LambdaMART	0.5306	0.5386	0.5199
ListNet	0.5543	0.5829	0.5783

and specify the vertical intent for the subtopics whose candidate vertical intent is Web. The detail of the method is described in the following sections.

2.4.1 Model Construction

Because we need to predict vertical intent for each subtopic query, it is important to select the queries which contain vertical intents when we build training dataset. After observing several of the most popular websites in China, we finally choose a Web directory[8] to generate our training queries. There are totally 15 categories and 83 subcategories. Each subcategory has a collection of related Web sites, including the name of Web sites and the corresponding urls. Table 3 shows an example of this Web directory. We can see that the Web sites in the same subcategories usually have similar search intent orientation, which can help us collect seed URLs that contain vertical intents. With these subcategories, we extract 1212 Web URLs covering the domain of news, videos, shopping, communities and games. Considering that click-through bipartite graphs reveal the strong connection between search queries and clicked results, we adopt a random walk algorithm to collect candidate queries starting from these seed URLs. Then these queries are used to crawl original search pages from a popular commercial search engine. We extract the distribution of vertical types based on the CSS styles of verticals. The aim of this step is to perform an automatic process to collect vertical intents of each candidate query. Note that a query may have multiple vertical intents. In order to cover enough potential vertical intents in our training dataset, we identify *presentation score* (P-score) for each vertical intents. It can be calculated as follows:

$$P\text{-}score = \begin{cases} \dfrac{1}{\log(1+R_i)} & \text{if exists} \\ 0 & \text{else} \end{cases} \quad (4)$$

where i denotes the type of a vertical, namely, Web, image, news, download, encyclopedia and shopping vertical, R_i is the highest ranking of the type i vertical on the original search pages. We suppose that the rank of vertical results reflect the level of correlation between the search query and vertical intent. If the vertical is placed in top positions on the search result page, the intent of this query is more likely to contain the type of this vertical. Therefore, according to the distribution of vertical results on search result pages, we calculate the presentation score of Web, image, news, download, encyclopedia and shopping vertical for each query, respectively. These scores are used as the target value for training our model. In order to represent the text of queries, all the candidate queries are calculated by 'word2vec' algorithm and changed into word vector presentation. The value of the query vector is the input feature in our model.

After generating the training dataset, we adopt a logistic regression algorithm to train our prediction models. For each type of verticals, we tend to train an independent model to predict the presentation score. By comparing the presentation score of different vertical types, we can select the verticals which gain the highest score as the candidate vertical intent. Therefore, we totally train six prediction models and use this strategy to predict the candidate vertical intents of our subtopic queries.

2.4.2 Vertical Diversification

In practical search scenarios, Web results occupy a large portions of search results and the Web intent may also cover multiple vertical intents. This may limit the performance of prediction models. To specify potential vertical intents in a subtopic, we use a rule-based method to diversify the vertical intents. First, we extract all the subtopic queries whose candidate intent is Web intent. Then we modify the intent based on the distribution of vertical results on search result pages. In order to reduce the noises caused by irrelevant verticals, we only focus on the vertical type of top 3 positions. If the vertical results are placed at the top of the search pages, we replace Web intent with the corresponding vertical intent. Moreover, the highest ranked vertical is selected as vertical intent if there are more than two verticals at top 3 position. As a result, we generate a mapping between subtopics and vertical intents.

2.5 Submitted Runs and Evaluation Results

We apply the methods described above to produce the following five runs for the Chinese Query Understanding Subtask. Vertical intents in all five runs are generated with the same algorithm described in Section 2.4

- **THUIR-QU-1A**: Adopt K-means algorithm to cluster all subtopic candidates into 10 clusters and select the candidate with the highest $S(q)$ from each cluster.
- **THUIR-QU-2A**: Adopt K-means algorithm to cluster all subtopic candidates into 5 clusters and select two candidates with the highest two $S(q)$ from each cluster.
- **THUIR-QU-1B**: Rerank the 10 subtopics generated by THUIR-QU-1A with learning to rank algorithm.
- **THUIR-QU-2B**: Rerank the 10 subtopics generated by THUIR-QU-2A with learning to rank algorithm.

[8]https://www.hao123.com

Table 2: An example of the Web directory

Category	Subcategory	URL of Web site	Description
Video	Movie	http://movie.youku.com/	An online video site
		http://www.iqiyi.com/dianying/	An online video site
	Television	http://cctv.cntv.cn/	The official Web site of CCTV
		http://www.brtn.cn/	The official Web site of Beijing Television Station
	News of Movie	http://ent.sina.com.cn/film/	A News Web site about Movies and Celebrities
		http://ent.163.com/	A News Web site about Movies and Celebrities

Table 3: Evaluation results for query understand subtask

	D♯-nDCG	*V-score*	*QU-score*
THUIR-QU-1A	0.5204	0.5579	0.5392
THUIR-QU-2A	0.5550	0.5506	0.5528
THUIR-QU-1B	0.5368	0.5763	**0.5565**
THUIR-QU-2B	0.5436	0.5686	**0.5561**
THUIR-QU-3A	0.4973	0.5942	0.5458

- **THUIR-QU-3A**: Adopt K-means algorithm to cluster all subtopic candidates into 5 clusters and select the candidate with the highest ten $S(q)$.

The *D♯-nDCG*, *V-score* and *QU-score* of the results are shown in Table 3. We can see that THUIR-QU-1B and THUIR-QU-2B perform the best in terms of *QU-score*. This is in line with our expectation because we assume that clustered candidates generated by K-means may contain some noises and a learning to rank algorithm can help reserve the ones of high quality. Between the two K-means clustering settings, runs with candidates clustered into 5 categories achieve a higher *D♯-nDCG* than those with candidates clustered into 10 categories. This may be because it is difficult to generate as many as 10 different subtopics for many queries. Thus some of the subtopics in THUIR-QU-1A and THUIR-QU-2A may be of low quality. We can also see that while THUIR-QU-3A achieves the lowest *D♯-nDCG*, it also gets the highest *V-score*. This may be because that predict vertical intents for diversified subtopics may be a difficult task and the Web vertical may be appropriate for most subtopics. As a result, if provided with a more diversified subtopic list, our vertical predicting algorithm will need to generate more various types of verticals other than the Web vertical, which may lead to a decrease in *V-score*.

3. CHINESE VERTICAL INCORPORATING

3.1 Retrieval Models for Organic Search Results

We first adopt the same probabilistic model based on BM25 model[3] and a word pair model[6] as well as retrieval strategies as the ones we used in NTCIR-9[4] to rank organic search results. All the retrieval processes are conducted on IMine-2 Chinese Web Corpus provided by organizers, which contains three different parts: title, content and hyperlink. These parameters used in our model are shown in Table 4.

$$R(q, D) = W_{BM25} + \alpha \times W_{wp} \tag{5}$$

Table 4: Parameters in retrieval models

part	α_1	k_1	b	w
Title	0.08	1.8	0.25	0.6
Content	0.2	0.6	0.35	0.2
Hyperlink	0.1	1.6	0.3	0.5

$$R(Q, D) = \sum_{i=1}^{10} R(q_i, D) \times S(q_i) \tag{6}$$

Considering that we have ten subtopics for each query, we calculate a relevance score for each subtopic and each document as shown in Equation 5, where q refers to a particular subtopic of the original query. Then in Equation 6, where $S(q_i)$ is defined in Equation 3 to measure the subtopic quality, we calculate the weighted sum relevance score for each query and document. We rank the documents by $R(Q, D)$ to maximum the quality as well as the diversity of the result list.

3.2 Vertical Ranking

Considering the virtual document of a vertical is always relevant to one of the intents behind the query, we use the relationship between subtopics and vertical intents, which has been generated in Query Understanding task, to incorporate these virtual verticals into ranked lists. We define a mapping function to evaluate the importance of the vertical documents. It can be calculated as,

$$I(v) = \alpha \cdot S\text{-}score(v) \tag{7}$$

where v denotes a type of vertical intent, *S-score* is the score of the subtopic which contains vertical intent in Query Understanding task. α is a variable parameter that reflects the connection between subtopic and the importance of virtual verticals. *I(v)* is a score that describes the importance of virtual documents in the ranked lists. By comparing the score of virtual verticals with organic documents, we incorporate the virtual verticals into the list of Web documents and rerank the documents. In the end, we generate a diversified search result list which contains multiple types of documents for each query.

4. CONCLUSIONS

In this paper, we introduce our approaches for NTCIR-12 IMine task. For the Chinese Query Understanding subtask, we try to deploy different strategies to mine candidate subtopics from various data resources. A two-step approach is adopted to predict vertical intents for each subtopic. For the Vertical Incorporating subtask, we adopt a probabilistic

retrieval model to rerank the lists of Web documents and further incorporate virtual verticals into these ranked result lists based on the result of Query Understanding subtask.

5. REFERENCES

[1] L. D. Catledge and J. E. Pitkow. Characterizing browsing strategies in the world-wide web. *Computer Networks and ISDN systems*, 27(6):1065–1073, 1995.

[2] C. Luo, X. Li, A. Khodzhaev, F. Chen, K. Xu, Y. Cao, Y. Liu, M. Zhang, and S. Ma. Thusam at ntcir-11 imine task. In *NTCIR*, 2014.

[3] S. Robertson, H. Zaragoza, and M. Taylor. Simple bm25 extension to multiple weighted fields. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 42–49. ACM, 2004.

[4] Y. Xue, F. Chen, T. Zhu, C. Wang, Z. Li, Y. Liu, M. Zhang, Y. Jin, and S. Ma. Thuir at ntcir-9 intent task. In *NTCIR*. Citeseer, 2011.

[5] T. Yamamoto, Y. Liu, M. Zhang, Z. Dou, K. Zhou, M. Ilya, M. P. Kato, H. Ohshima, and F. Sumio. Overview of the ntcir-12 imine task. In *NTCIR*, 2014.

[6] M. Zhang, C. Lin, Y. Liu, L. Zhao, and S. Ma. Thuir at trec 2003: Novelty, robust and web. In *TREC*, pages 556–567, 2003.