ICTNET at Federated Web Search Track 2014

Feng Guan[123], Shuiyuan Zhang[123], Chunmei Liu[123], Xiaoming Yu[12], Yue Liu[12], Xueqi Cheng[12]

1)Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190

2)Key Laboratory of Web Data Science and Technology,CAS

3) University of Chinese Academy of Sciences, Beijing, 100190

{ guanfeng,zhangshuiyuan,liuchuanmei,yuxiaoming }@software.ict.ac.cn; {liuyue,cxq}@ict.ac.cn

1. Introduction

We have participated all the three tasks of FedWeb 2014 this year. Basic methods that we used for these tasks will be described in section 2. Section 3 shows combination of the basic methods for different runs and the results will also be introduced.

2. Proposed methods

2.1 Vertical Selection task

2.1.1 LSI model

For a given query, vertical selection will choose a subset of candidate verticals to retrieve from. The most intuitive approach is to look for the most similar query vertical pairs. To find a representation for both vertical and query, we take the help of Google Custom Search API (GCSA)[1]. With this API we can build a simple Custom Search Engine, and 10 results for each query will be returned as query representation. We also use a small portion of the given documents as vertical representation.

After query and vertical representation, we choose Latent Semantic Index (LSI) model to calculate similarity between them. LSI model uses a mathematical technique called singular value decomposition (SVD) to identify patterns in the relationships between terms [3]. As our representations of query and vertical are not likely similar in a literal way, LSI model can help us find the hidden commonality between vertical and query. Then two steps of voting will be taken. First, similarity between vertical representations and query representations will be calculated and the top 40 most similar vertical representations will be scored based on their similarity ranking. The score of vertical representation came from the same vertical will be summed. For each query representation, verticals whose score are no less than the threshold will be selected to the next voting step. Second, for each query, we choose verticals that appear more than twice as the candidate vertical set of all this query representations.

2.1.2 Text Classification

We use the state-of-art algorithm Random Forest (RF) to do classify job. RF model will be trained using the given documents. Each vertical is a class. Both the number of trees and the number of features in random feature selection are 100. We use tf-idf of terms as the features of documents, and do query expansion with GCSA based on the top 10 relevant documents. For each document returned, we can get its probabilities classified to a vertical by RF. Then, we sum the probabilities of the 10 documents as vertical's score. Finally, we recommend the verticals that have high score.

2.1.3 Frequent Term Rank(FTR) based vertical representation

We believe that, excluding stop words, the more frequent that a term is in one vertical, the more probable that it can represent this vertical. Besides, the order of frequent terms maybe different during verticals. Thus we obtain the similarity score of query Q and vertical V according to equation 1 as follow:

$$sim(Q,V) = \sum_{q \in Q} \frac{1}{Rank(q,V)} \tag{1}$$

Rank(q, V) means the rank of term q in feature vector of vertical V.

2.2 Resource Selection task

2.2.1 LSI model

LSI model used in resource selection task is similar with vertical selection task. Resource is represented by 20 random documents from provided data at this resource. Considering authority of resource will have an effect on ranking, we collect page rank score for every resource [2]. Besides, each resource belongs to one vertical, vertical ranking also has a huge impact on resource selection.

The final similarity score of a query resource pair is given in Equation 2. We choose top 20 resources for each query.

$$score_i = \sum (lsi\_sim + \frac{1}{1 + \alpha \times vertical\_rank_i} + pr \times pr\_weight) \tag{2}$$

2.2.2 Information Retrieval

We build index of the provided documents with stop words. For each query and each resource, we sum the scores of documents in one resource as the score of one resource. Each field has different weight. When searching, we combine the results of query without stop words and the same query within window size of 5. Each document's final score should be multiplied by Penalty Factor. The Penalty Factor is defined as equation 3, which α equals 0.5, and R represents the ranking of each document.

$$penalty\_factor = \frac{1}{1 + \alpha \times R} \tag{3}$$

2.2.3 Text Classification

This method is similar to that used in VS task. The difference is that the classes are the resources. Besides, the score of each resource should relate to the rank of resources in VS task. Therefore, we raise the score of those resources.

2.3 Result Merging task

2.3.1 LSI model

Like previous tasks, query is represented by 10 different snippets with GCSA. For each snippet, we use LSI model to calculate similarity between query representation (snippet) and document. For each query representation, snippet is more important if it has a higher rank in search results. So we define google_weight as Equation 4. We also take page rank of page host into account in this task.

$$google\_weight = \frac{1}{1 + \alpha \times rank} \tag{4}$$

In the provided data, a document can appear in many resources with same url and different doc id. The more search engines a document appears in, the more important the document is. Extra score will be added to documents which occur many times in different resources.

The final score for a document *i* is defined as Equation 5, pr_weight is used to adjust page rank to the same scale.

$$score_i = \sum (lsi\_sim \times google\_weight + pr \times pr\_weight + extra\_score) \tag{5}$$

2.3.2 Information Retrieval

We get origin score using information retrieval method mentioned above. We use the results in

RS task and the origin score is multiplied by the penalty factor as we did in the RS task. What's more, we considered the duplicated times of each document and the ranking of the resource which the document belongs to. The expression defined as equation 6:

$$score_2 = score_1 \times \frac{1}{1+\alpha \times R_0} \times \sum_0^{dup} \frac{1}{1+\alpha \times R_{r,i}} \tag{6}$$

Which the "*dup*" is the duplicated times of one document, and $R_{r,i}$ represents the document i's resource's ranking in last task.

2.3.3 Pagerank-like model

Like the algorithm of PageRank, for each query, we set the initial Pr values as the scores generated by Information Retrieval method. The weight of each edge is the similarity generated by LSI model. The expression defined as equation 7:

$$pr_i^j = \alpha + (1-\alpha)\sum_{k=0}^{I_i} \frac{pr_k^{j-1} \times sim_{ik}}{O_K} \tag{7}$$

Which pr_i^j represents the *PR* value of document *i* in the *jth* iteration. I_i represents the number of ingoing links of document *i*, O_k represents the outgoing links of document *k*, sim_{ik} represents the LSI similarity between document *k* and *i*, the *α* is Damping Factor equals 0.85.When the *PR* values tend to be stable, iteration would stop. The PR value of each document will be the final score of each document.

2.4. Ensemble methods

In machine learning area, ensemble methods using multiple learning algorithms could obtain better predictive performance than any of the constituent learning algorithms [4]. We use ensemble methods in vertical selection and result merge task to merge our single model results, and for resource selection task, our submission run only based on single model result.

For vertical selection task, our ensemble method is to try getting a higher recall in single model, and intersecting single model results to get a balance precision and recall result. For result merge task, different model may have different score scale, we only use ranking data. Given several single model results, final score for a document i defined as Equation 8. Rank in the same interval will have same score, we choose interval=3 in our experiment.

$$score_i = \sum \frac{1}{1.0 + \alpha \times (rank_i / interval)} \tag{8}$$

3. Results

3.1 Vertical Selection task

ICTNETVS1 is based on traditional information retrieval (IR) model.

ICTNETVS02 uses Random Forest text classification model, the result is the sum of probabilities.

ICTNETVS03 is based on LSI model using vertical representation and query representation with GCSA.

ICTNETVS04 uses ensemble method with 2 LSI models and text classification model.

ICTNETVS05 combines 2 LSI models and text classification model and information retrieval model together using set intersection.

ICTNETVS06 uses Random Forest text classification model, the result is the sum of voting.

ICTNETVS07 is the Borda Fuse combination of three methods. The first one is to calculate similarity between vertical and query using FTR. The second one also uses FTR that co-occurrence terms are used to expand query. The third one is IR model.The result is given below.

Table 1: Vertical Selection results(+- st.dev.)

Runtag	Precision	Recall	F1-measure
ICTNETVS1	0.230 (+-0.202)	0.638 (+-0.407)	0.299 (+-0.201)
ICTNETVS02	0.292 (+-0.201)	0.790 (+-0.368)	0.401 (+-0.228)
ICTNETVS03	0.276 (+-0.298)	0.410 (+-0.436)	0.298 (+-0.303)
ICTNETVS04	0.427 (+-0.419)	0.392 (+-0.419)	0.377 (+-0.375)
ICTNETVS05	0.423 (+-0.441)	0.365 (+-0.417)	0.359 (+-0.381)
ICTNETVS06	0.258 (+-0.201)	0.673 (+-0.394)	0.344 (+-0.217)
ICTNETVS07	0.591 (+-0.411)	0.545 (+-0.391)	0.496 (+-0.337)

3.2 Resource Selection task

ICTNETRS01 uses traditional information retrieval model.

ICTNETRS02 and ICTNETRS07 take VS results into account while using information retrieval model. The difference is we use different vertical selection result.

ICTNETRS03 uses text classification model (RF), meanwhile VS results are considered.

ICTNETRS04 uses LSI model with page rank.

ICTNETRS05 and ICTNETRS06 use LSI model with page rank and vertical selection results.

The result is given below.

Table 2: Resource Selection results(+- st.dev.)

Runtag	nDCG@20	nDCG@10	nP@1	nP@5
ICTNETRS01	0.268 (+-0.147)	0.226 (+-0.162)	0.163 (+-0.269)	0.193 (+-0.171)
ICTNETRS02	0.365 (+-0.154)	0.322 (+-0.175)	0.289 (+-0.334)	0.324 (+-0.197)
ICTNETRS03	0.400 (+-0.123)	0.340 (+-0.138)	0.160 (+-0.276)	0.351 (+-0.165)
ICTNETRS04	0.362 (+-0.113)	0.306 (+-0.146)	0.116 (+-0.256)	0.290 (+-0.212)
ICTNETRS05	0.436 (+-0.149)	0.391 (+-0.173)	0.489 (+-0.391)	0.377 (+-0.194)
ICTNETRS06	0.428 (+-0.160)	0.372 (+-0.176)	0.521 (+-0.377)	0.345 (+-0.197)
ICTNETRS07	0.373 (+-0.143)	0.334 (+-0.171)	0.267 (+-0.338)	0.334 (+-0.196)

3.3 Result Merging task

All of all submissions of result merging task are based on the provided resource selection baseline, this baseline is the same as our ICTNETRS06.

ICTNETRM01 uses information retrieval method.

ICTNETRM02 removes duplicate urls from ICTNETRM01.

ICTNETRM03 uses Pagerank-like model, while the similarity is calculated with LSI model without duplicate urls.

ICTNETRM04 is based on LSI model, topic number is 5.

ICTNETRM05 uses 3 models ensemble method, which are LSI model, IR model and PR model.

ICTNETRM06 removes duplicate urls from ICTNETRM05.

ICTNETRM07 uses 2 models ensemble method, which are IR model and PR model, and duplicate urls are also removed.Detailed results are shown as table 3. Ensemble method without duplicate urls gets the best score.

Table 3: Result Merging results(+- st.dev.)

Runtag	nDCG@20	nDCG@20_withdup	nDCG@20_local	nDCG-IA@20
ICTNETRM01	0.247 (+-0.146)	0.361 (+-0.215)	0.338 (+-0.186)	0.080 (+-0.054)
ICTNETRM02	0.309 (+-0.174)	0.314 (+-0.181)	0.362 (+-0.182)	0.095 (+-0.059)
ICTNETRM03	0.348 (+-0.160)	0.350 (+-0.161)	0.405 (+-0.158)	0.111 (+-0.063)
ICTNETRM04	0.381 (+-0.157)	0.386 (+-0.157)	0.451 (+-0.142)	0.121 (+-0.063)
ICTNETRM05	0.354 (+-0.138)	0.492 (+-0.201)	0.497 (+-0.183)	0.123 (+-0.071)
ICTNETRM06	0.402 (+-0.153)	0.407 (+-0.159)	0.473 (+-0.138)	0.132 (+-0.070)
ICTNETRM07	0.386 (+-0.153)	0.390 (+-0.157)	0.451 (+-0.148)	0.123 (+-0.068)

4. Acknowledgement

We would like to thank all organizers and assessors of TREC and NIST. This work is sponsored by 973 Program of China Grants *No.2012CB316303&No.2014CB340405*, 863 program of China Grants *No.2012AA011003&No.2014AA015204*, NSF of China Grants *No.61232010&No.61173064,* and by the National Key Technology R&D Program Grants *No. 2012BAH39B02&No. 2012BAH39B04.*

Reference

[1] https://developers.google.com/custom-search/.

[2] http://www.prchecker.pw/bulk-pagerank-checker.php.

[3] Scott Deerwester. Improving information retrieval with latent semantic indexing. 1988.

[4] Richard Maclin and David Opitz. Popular ensemble methods: An empirical study. arXiv preprintarXiv:1106.0257, 2011.