

# Learning from Past Queries for Resource Selection

Suleyman Cetintas, Luo Si, Hao Yuan  
Department of Computer Sciences, Purdue University  
West Lafayette, IN, 47907, USA  
{scetinta,lsi,yuan3}@cs.purdue.edu

## ABSTRACT

Federated text search provides a unified search interface for multiple search engines of distributed text information sources. Resource selection is an important component for federated text search, which selects a small number of information sources that contain the largest number of relevant documents for a user query. Most prior research of resource selection focused on selecting information sources by analyzing *static* information of available information sources that is sampled in the offline manner. On the other hand, most prior research ignored a large amount of valuable information like the results from past queries.

This paper proposes a new resource selection technique (which is called *qSim*) that utilizes the search results of *past queries* for estimating the utilities of available information sources for a specific user query. Experiment results demonstrate the effectiveness of the new resource selection algorithm.

## Categories and Subject Descriptors

H.3.3 [Information Search and Retrieval]:

## General Terms

Algorithms

## Keywords

Past Queries, Resource Selection, Federated Search

## 1. INTRODUCTION

The proliferation of searchable text information sources on local area networks and the Internet creates a problem of finding information that is distributed among many text information sources (federated text search) [1,3]. One of the main topics of federated search is *resource selection* or *collection selection*, which is the task for selecting a small number of text information sources for search for a specific user query [3,5,6,7].

Most prior research work of resource selection focused on selecting information sources by analyzing the static information which is sampled from available information sources in an offline manner. For example, the ReDDE (Relevant Document Distribution Estimation [7]) algorithm first tries to build a centralized sampled database based on query-based-sampling, and then uses the information of the centralized sampled database to estimate the distribution of relevant documents for resource

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CIKM'09, November 2–6, 2009, Hong Kong, China.

Copyright 2009 ACM 978-1-60558-512-3/09/11...\$10.00.

selection. However, such approaches ignored a large amount of valuable information like the resource selection results of past queries. For instance, in a real world search engine, there are many similar or even duplicated queries every day. The results from those similar past queries are very valuable to guide the resource selection decision of a current user query.

In this paper, we propose a new resource selection technique (i.e., *qSim*) that utilizes the search results of past queries for estimating the utilities of available information sources for a specific user query. The new algorithm calculates the query similarities between a specific query and all past queries, and then estimates the utilities of available information sources by the weighted combination of search results of past queries with respect to the query similarities. The new resource selection algorithm is practical as it does not require relevance judgment of past queries, and it only utilizes regression based results merging method to rank the results of past queries.

## 2. PRIOR RESEARCH

This section briefly surveys related work in resource selection. There is a large body of prior research on resource selection (e.g., [3,5,6,7]). Space limitations preclude discussing it all, so we restrict our attention to a few that have been studied often or recently in prior research.

A large body of resource selection algorithms follows the “Big Document” approach by representing each information source as a single big document. The similarities between the “Big Documents” and a user query are calculated to rank available information sources. Particularly, the CORI resource selection algorithm [1] has been shown to be one of the most stable and effective “Big Document” resource selection algorithms in prior studies (e.g., [3]). However, the “Big Document” approach does not consider information source sizes, which is a big problem for searching in the environment of a mixture of “small” and “very large” information sources [7].

ReDDE [7] resource selection algorithm was proposed to address the above issue. It explicitly estimates the distribution of relevant documents among available information sources for resource selection. ReDDE utilizes database size estimation and a centralized sample database (CSDB) that consists of the documents obtained by query based sampling. The CSDB is a representative subset of the centralized complete database (CCDB) which is the union of all the documents in available information sources. Since the CCDB is not available in the federated search environment, ReDDE uses the CSDB to simulate the property of CCDB. Details of ReDDE algorithm can be found in [7].

However, all the above prior research of resource selection utilized only static information of available information sources and ignored a large amount of valuable information like the

results from past queries. In most real world applications, there is often substantial similarity between users' queries; and even many of them are duplicates of each other. This fact motivates the research to make use of the results of past queries.

Voorhees et al.'s work considered using results of past queries [9]. However, it is not exactly resource selection since it retrieves documents from all information sources. Furthermore their methods require relevance judgment.

### 3. NEW RESOURCE SELECTION ALGORITHMS

Since there tends to be many similar queries in a real world federated search system, the valuable information of past queries can help us provide better resource selection results. In this section, we propose a novel algorithm, which is called *qSim*, to utilize the valuable information to guide the decision of resource selection.

Assume that there exists a set of past queries, which is denoted by  $P = \{p_1, p_2, \dots, p_M\}$ , where  $p_i$  represents the  $i^{\text{th}}$  past query. For an online user query  $q$ , assume that we have a specific method to measure the similarity between  $q$  and  $p_i$  (we will further discuss this in section 3.2), let us denote the similarity by  $Sim(p_i | q)$ .

Denote the set of information sources by  $S = \{s_1, s_2, \dots, s_N\}$ . For a query  $q$ , we use  $Rel(s_j | q)$  to represent how likely it is for the information source  $s_j$  to be relevant to the query  $q$ , or what (estimated) percentage of the relevant documents for the query  $q$  is in  $s_j$ .

In *qSim*, we can estimate the value of  $Rel(s_j | p_i)$  for all past queries. For an online query  $q$ , the task is to estimate  $Rel(s_j | q)$  based on the information of  $Rel(s_j | p_i)$  and  $Sim(p_i | q)$ .

#### 3.1 Estimating $Rel(s_i | p_i)$

Any resource selection algorithm can be applied to estimate the value of  $Rel(s_j | p_i)$ . If a resource selection algorithm can explicitly give scores for all information sources, we can directly map those scores to  $Rel(s_j | p_i)$ . Here, we present a regression based approach to get a much more precise  $Rel(s_j | p_i)$  estimation by utilizing the search results from past queries, no matter whether a prior resource selection has been made for  $p_i$  or not.

First, assume that a set of sources (denoted by  $SEL_{p_i}$ ) are selected for  $p_i$ . For the case that no prior resource selection has been made, we can have  $SEL_{p_i} = S$ .

Second, we use the regression based Semi-Supervised Learning method [5] to merge the search results from those sources in  $SEL_{p_i}$  into a single ranked list.

Third, we assign scores for the information sources by the following way: If the source  $s_j$  is not in  $SEL_{p_i}$ , then  $Rel(s_j | p_i) = 0$ ; If the source  $s_j$  is in  $SEL_{p_i}$ , then we estimate  $Rel(s_j | p_i)$  by:

$$Rel(s_j | p_i) \propto \sum_{\text{top } T \text{ docs in the merged ranked list}} \frac{Rel(s_j | \text{doc})}{T} \quad (1)$$

where  $T$  is a pre-defined number, and

$$Rel(s_j | \text{doc}) = \begin{cases} 1, & \text{if doc} \in s_j; \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

The value  $T$  can be very small like 20; and we set it to 20.

Please note that when we build the regression models for merging the search results, some documents need to be downloaded for each past query. To evaluate the resource selection approach of learning from past queries in a strict manner, we do not allow it to obtain documents in the centralized sample database as training documents to build regression models. On the other side, we download the top 3 documents to build the regression models for results merging in a similar manner as the minimum downloading method of the Semi-Supervised Learning results merging algorithm [5].

#### 3.2 Estimating $Sim(p_i | q)$

There are many ways to measure the similarities between queries, such as term-based approach (edit distance, latent semantic analysis, etc), selection-based approach and result-based approach. In our experiments, we use a result-based approach to calculate the similarities.

Assume that there is a database which contains relevant information to our queries. One way to construct such a database is to index the documents that are downloaded when we build regression models for merging search results [8] for past queries.

Let  $R_{p_i}$  (or  $R_q$ ) denote the ranked list of returned documents when we search  $p_i$  (or  $q$ ) in the downloaded documents database. We measure the similarity of  $p_i$  with respect to  $q$  by

$$Sim(p_i | q) \propto \frac{1}{|R_{p_i}|} \sum_{\text{doc} \in R_{p_i} \cap R_q} Score(\text{doc}, R_{p_i}, R_q) \quad (3a)$$

where the score function for a matching document is

$$Score(\text{doc}, R_{p_i}, R_q) = 1 - \left| \frac{\text{doc rank in } R_{p_i}}{|R_{p_i}|} - \frac{\text{doc rank in } R_q}{|R_q|} \right| \quad (3b)$$

In practice, we cut each search results ( $R_{p_i}$  and  $R_q$ ) by only focusing on the top 20% returned documents in their rank lists.

Finally, we normalize the values of  $Sim(p_i | q)$  according to the following formulas.

$$\begin{aligned} MAXSIM_q &= \max_i Sim(p_i | q) \\ CUTSIM_q &= 0.8 * MAXSIM_q \end{aligned} \quad (4)$$

$$\text{normalized } Sim(p_i | q) \propto \begin{cases} 0, & \text{if } Sim(p_i | q) < CUTSIM_q \\ \frac{(Sim(p_i | q) - CUTSIM_q)}{(MAXSIM_q - CUTSIM_q)}, & \text{otherwise} \end{cases} \quad (5)$$

#### 3.3 Predicting $Rel(s_i | q)$

For an online query  $q$ , we predict  $Rel(s_j | q)$  based on estimated relevant information from similar past queries. It is calculated by the following formula

$$Rel(s_j | q) \propto \sum_i Rel(s_i | p_i) Sim(p_i | q) \quad (6)$$

In practice, we only consider the  $K$  most similar past queries when we calculate the above summation. In our experiments,  $K$  is set to 5.

### 3.4 Resource Selection According to $Rel(s_i | q)$

The last step is to simply rank the information sources according to the values of  $Rel(s_j | q)$ . A larger value of  $Rel(s_j | q)$  means that it is more likely for the source  $s_j$  to contain more relevant information with respect to  $q$ .

### 3.5 Different Levels of Past Search Results

For a past query  $p_i$ , the amount of search results will affect the quality of the estimated  $Rel(s_j | p_i)$ . The amount of results is measured by the number of information sources selected and searched for generating search results. If more sources have been searched, the amount of search results for the past query is more comprehensive.

We call the qSim algorithm by “ $qSim-Cut-X$ ” where the number of sources selected and searched for  $p_i$  is  $X$ , or in other words  $|SEL_{p_i}| = X$ . In a real world federated search application with  $N = 100$  information sources, the typical number of sources that are chosen for the search task is about  $X = 5, 10$  or at most 20. In this work,  $X$  is chosen to be 10 for efficiency purposes.

## 4. EXPERIMENTAL METHODOLOGY

An extensive set of experiments was designed in research environments to simulate real world applications.

### 4.1 Testbeds

Experiments were carried out on Trec123 and Trec4 testbeds. Details about Trec123 and Trec4 datasets are given in Table 1.

**Trec123-100col-bysource (Trec123):** 100 information sources were created from TREC CDs 1, 2 and 3. They are organized by source and publication date.

**Trec4-bysource (Trec4):** 100 information sources were created from TREC4 according to the sources of the documents in TREC4.

For Trec123 and Trec4 testbeds, 50 queries were created from the title fields of TREC topics 51-100 and from the description fields of TREC topics 201-250 respectively. These queries will be used as the set of test/online queries. The detailed statistics about the test queries can be found in [7].

All the 100 information sources were assigned one of three types of retrieval algorithms as INQUERY [2], a unigram language model with linear smoothing [4] and a TFIDF retrieval algorithm with the “ltc” weighing schema [8]. All the algorithms were implemented with the lemur toolkit [4].

### 4.2 Sampled and Simulated Past Query Sets

In a real world federated search system, there often exist many similar queries. However in Trec123 and Trec4 testbeds, we don’t have many similar queries. Therefore we use two different approaches to generate two different sets of past queries using the test queries that are available in Trec123 and Trec4 datasets. Then the set of real queries that are available from Trec123 and Trec4

**Table 1: Summary statistics for Trec123 and Trec4 distributed IR testbeds.**

Testbed	Size (GB)	Number of Documents (x1000)			Size (MB)		
		Min	Avg	Max	Min	Avg	Max
Trec123	3.2	0.7	10.8	39.7	28	32	42
Trec4	2.0	5.6	5.6	5.6	4	20	138

datasets are used as the set of test/online queries and either one of the two sets of generated past queries are used as the set of past queries.

#### 4.2.1 Sampled Past Queries

The first approach of generating the set of past queries samples queries for each test/online query by extracting the titles of some top ranking documents in the search results of that particular test query. The past queries generated by this approach will be referred as *sampled past queries*. One set of sampled past queries is generated for Trec123, which is called Trec123\_T20. The past queries in Trec123\_T20 were generated in the following way:

- A new Trec123 database called Sub\_Trec123 is constructed, which consists of the documents that exist in the original Trec123 database and that do not exist in the CSDB of the ReDDE algorithm.
- For each test query, 20 most similar documents that have a title (not all the documents have a title) are retrieved from Sub\_Trec123 database. The titles of these TOPX documents construct the set of 20 sampled queries for this test query.

For 50 test queries, the above sampling process is repeated and a total of 1000 past queries are acquired for Trec123\_T20. One set of sampled past queries is also generated for Trec4 testbed in the same way described above. The average number of words in Trec123\_T20 and Trec4\_T20 query sets are 8.98 and 8.53 respectively.

#### 4.2.2 Simulated Past Queries

The second approach of generating the set of past queries creates the past queries from test queries by randomly removing some terms from the test queries. The past queries generated by this approach will be referred as *simulated past queries*. One set of past queries is generated for each of Trec123 and Trec4 testbeds, which are called Trec123\_R1 and Trec4\_R2. The past queries in Trec123\_R1 and Trec4\_R2 were generated in the following way:

- For each test query, 1 (for Trec123\_R1) or 2 (for Trec4\_R2) term(s) is (are) removed to generate a simulated past query;
- For each simulated past query at least 2 (for Trec123\_R1) or 3 (for Trec4\_R2) terms are kept to make sure the simulated past query is not too short to be meaningful.

In the experiments that used the simulated past queries as the set of past queries; the 50 real queries are treated as test/online user queries, and the 50 simulated queries (i.e. in Trec123\_R1 or Trec4\_R2) are treated as past queries. The average number of words in Trec123\_R1 and Trec4\_R2 are 2.60 and 6.92 respectively. Since the processes of generating the simulating queries include randomness, each set of simulated queries (e.g., Trec123\_R1) has been generated for 15 times. Accordingly, the resource selection experiments that use simulated past queries are run 15 times for each set of simulated queries and the average results of all of these runs are reported for evaluation.

**Table 2: Comparison between ReDDE with qSim-Cut-10 on Trec123 dataset with Trec123\_T20 sampled past query set and with Trec123\_R1 simulated past query set.**

# Selected Sources	ReDDE on Trec123	qSim-Cut-10 on Trec123_T20	qSim-Cut-10 on Trec123_R1
1	0.2445	0.3414 (39.66%)	0.4003 (63.71%)
2	0.3121	0.3675 (17.73%)	0.4229 (35.47%)
3	0.3188	0.4021 (26.15%)	0.4242 (33.06%)
4	0.3350	0.3929 (17.31%)	0.4372 (30.53%)
5	0.3503	0.4108 (17.26%)	0.4410 (25.90%)
6	0.3745	0.4093 (9.30%)	0.4336 (15.79%)
7	0.3889	0.4194 (7.84%)	0.4356 (12.02%)
8	0.4025	0.4373 (8.66%)	0.4392 (9.15%)
9	0.4108	0.4429 (7.83%)	0.4454 (8.43%)
10	0.4325	0.4577 (5.83%)	0.4461 (3.14%)
Overall Improvement		15.76%	23.72%

### 4.3 Baseline Method

In the experiments, we use ReDDE to do resource selection for past queries (i.e. to determine  $SEL_{p_i}$ ). Also, ReDDE is a baseline resource selection algorithm to compare with qSim. For ReDDE, we build a centralized sampled database, which contains 150 documents per source by query-based sampling. All other parameters in ReDDE are the same as that in [7].

### 4.4 Evaluation Metric

The recall metric  $R_k$  has been commonly used to evaluate resource selection algorithms [1,7]. It measures what percentage the difference is between the estimated ranking and the most desirable ranking. Therefore, at a fixed  $k$ , a larger  $R_k$  value indicates a better ranking.

## 5. EXPERIMENT RESULTS

Experiments are conducted to address how good qSim is in comparison to the state-of-the-art ReDDE algorithm.

Table 2 and Table 3 show the results of qSim and ReDDE on the Trec123 and Trec4 testbeds with Trec123\_T20 and Trec4\_T20 sampled query sets and Trec123\_R1 and Trec4\_R2 simulated query sets. The performance is evaluated by the Recall metric  $R_k$  defined in section 4.4, where “# Selected Sources” in the tables is the  $k$  in  $R_k$ . The percentages within the parentheses are the relative improvements of qSim over ReDDE. The overall improvement is calculated by taking the average of the improvement percentages when 1,2,...,10 sources are selected.

As shown from Table 2 and Table 3, qSim-Cut-10 always generates much better results than ReDDE. Thus, qSim can be considered as a very effective algorithm for resource selection.

## 6. CONCLUSION

Resource selection is an important component for a federated text search system. Prior research of resource selection ignored a large amount of information of the results from past queries.

This paper proposes a new resource selection approach called qSim to utilize the search results of past queries for estimating the utilities of available information sources for a specific user query. For a user query, the algorithm first calculates the similarity measurements between the current user query and the past queries. The qSim algorithm then estimates the utilities of available information sources by the weighted combination of search results of past queries with respect to the query similarity

**Table 3: Comparison between ReDDE with qSim-Cut-10 on Trec4 dataset with Trec4\_T20 sampled past query set and Trec4\_R2 simulated past query set.**

# Selected Sources	ReDDE on Trec4	qSim-Cut-10 on Trec4_T20	qSim-Cut-10 on Trec4_R2
1	0.2739	0.3221 (17.65%)	0.2882 (05.23%)
2	0.3375	0.3288 (-02.56%)	0.3065 (-09.20%)
3	0.3031	0.3295 (08.73%)	0.3176 (04.80%)
4	0.3101	0.3499 (12.85%)	0.3306 (06.62%)
5	0.3095	0.3590 (16.01%)	0.3447 (11.37%)
6	0.3247	0.3623 (11.60%)	0.3554 (09.45%)
7	0.3339	0.3654 (09.44%)	0.3665 (09.76%)
8	0.3599	0.3681 (02.29%)	0.3788 (05.24%)
9	0.3753	0.3678 (-01.96%)	0.3896 (03.83%)
10	0.3905	0.3767 (-03.52%)	0.3972 (01.74%)
Overall Improvement		7.05%	4.88%

measurements. The qSim algorithm does not require relevance judgment of past queries and only uses the ranked lists of past queries, which are generated by regression based results merging method. Experiment results have demonstrated the advantage of qSim to achieve more effective results in comparison to the ReDDE algorithm.

## 7. ACKNOWLEDGEMENTS

This research was partially supported by the NSF grant IIS-0746830. Any opinions, findings, conclusions, or recommendations expressed in this paper are the authors', and do not necessarily reflect those of the sponsor.

## 8. REFERENCES

- [1] J. Callan. (2000). Distributed information retrieval. In W.B. Croft, editor, Advances in Information Retrieval. Kluwer Academic Publishers. (pp. 127-150).
- [2] J. Callan, W.B. Croft, and J. Broglio. (1995). TREC and TIPSTER experiments with INQUERY. Information Processing and Management, 31(3). (pp. 327-343).
- [3] N. Craswell. (2000). Methods for distributed information retrieval. Ph. D. thesis, The Australian Nation University.
- [4] The lemur toolkit. <http://www.cs.cmu.edu/~lemur>
- [5] A.L. Powell, J.C. French, J. Callan, M. Connell, and C.L. Viles. (2000). The impact of database selection on distributed searching. In Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.
- [6] M. Shokouhi and J. Zobel. (2007). Federated text retrieval from uncooperative overlapped collections. In Proceedings of the 30th Annual international ACM SIGIR Conference on Research and Development in information Retrieval
- [7] L. Si and J. Callan. (2003). Relevant document distribution estimation method for resource selection. In Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.
- [8] L. Si and J. Callan. (2003). A Semi-Supervised learning method to merge search engine results. ACM Transactions on Information Systems, 21(4). (pp. 457-491).
- [9] E. Voorhees, N. K. Gupta, and B. J. Laird. (1995). Learning collection fusion strategies. In Proc. of the 18<sup>th</sup> Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.