2011

Using Past Queries for Resource Selection in Distributed Information Retrieval

Sulleyman Cetintas
Purdue University, scetinta@cs.purdue.edu

Luo Si
Purdue University, lsi@cs.purdue.edu

Hao Yuan
Purdue University, yuan3@cs.purdue.edu

Report Number:
11-012

Using Past Queries for Resource Selection in Distributed Information Retrieval

Suleyman Cetintas[1] SCETINTA@CS.PURDUE.EDU
Department of Computer Sciences
Purdue University
West Lafayette, IN, 47907, USA

Luo Si LSI@CS.PURDUE.EDU
Departments of Computer Sciences and Statistics
Purdue University
West Lafayette, IN, 47907, USA

Hao Yuan YUAN3@ CS.PURDUE.EDU
Department of Computer Sciences
Purdue University
West Lafayette, IN, 47907, USA

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. 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 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. Furthermore, a *combined resource selection approach* is proposed to integrate the two approaches of learning from past queries and using static sampled information. An extensive set of experiments demonstrate the effectiveness of the new resource selection algorithm of learning from past queries as well as the combined resource selection algorithm in several configurations.

Keywords: Past Queries, Resource Selection, Federated Search

[1] Corresponding Author: Phone: (765) 494-9165 & Fax: (765) 494-0739

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) (Callan 2000; Craswell 2000; Meng et al. 2002). Federated text search, also known as distributed information retrieval, includes three sub-problems. First, information about the contents of each individual information source must be acquired. This task is called *resource representation* (Baillie et al. 2009; Callan 2000; Gravano et al. 1997; Nottelmann and Fuhr 2003; Si and Callan 2003a). Second, a small number of text information sources should be selected for search for a specific user query (Cetintas et al. 2009; Craswell 2000; French et al. 1999; Fuhr 1999; Gravano et al. 1999; Hawking and Thistlewaite 1999; Ipeirotis and Gravano 2002; Nottelmann and Fuhr 2003; Powell et al. 2000; Shokouhi 2007; Shokouhi and Zobel 2007; Si and Callan 2003a; Si and Callan 2004; Zhai and Lafferty 2001). This task is called *resource selection* or *collection selection*. Third, after results are returned from selected information sources, the individual ranked lists should be merged into a single final ranked list. This task is called *results merging* (Callan 2000; Cetintas and Si 2007; Kirsch 1997; Larson 2002; Le Calv and Savoy 2000; Lu et al. 2005; Si and Callan 2003b; Xu and Callan 1998).

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 (Si and Callan 2003a)) 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 selection.

However, most previous research ignored a large amount of valuable information like the resource selection results of past queries. For example, 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. One can imagine that if two user queries are very similar, their corresponding resource selection results should also be close.

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.

Furthermore, a combined resource selection approach is proposed to integrate the two types of approaches: learning from past queries and using static sampled information. It has its advantages to integrate two types of evidence. In particular, the combined approach generates the resource selection results by integrating the results of the qSim selection algorithm and the ReDDE selection algorithm.

Empirical studies are conducted on two testbeds with several configurations to show the advantage of the new resource selection algorithms. In particular, we compare the performance of the new algorithms with a state-of-the-art resource selection algorithm as ReDDE (Si and Callan 2003a) (relevant document distribution estimation). In our experiments in research environments, we followed two different approaches to generate the set of past queries: i) we sampled queries for each test/online query by extracting the titles of some top ranking documents in the search results of that particular test query; ii) we generated simulated queries by randomly removing some terms from test queries, and used these simulated queries as past queries. The set of all sampled queries and simulated queries for each test query forms the set of similar past queries in either approach. Under the usual scenario that there exist similar queries among past queries and online queries, our new qSim algorithm performs better than the prior state-of-the-art ReDDE algorithm. The larger amount of past queries we have or the more similar the past queries are to online queries, the better results we can get. Furthermore, a combined resource selection approach can provide even better resource selection results than both the qSim and the ReDDE algorithms. Therefore when the system receives a new rare test/online query, it will utilize the combined approach and perform at least as good as the ReDDE algorithm or even better than that.

The rest of the paper is arranged as follows: Section 2 discusses the related work. Section 3 proposes the new resource selection approaches, i.e. i) the resource selection approach by learning from past queries and ii) the combined resource selection approach. Section 4 discusses the experimental methodology. Section 5 presents the experimental results; and finally, Section 6 concludes this work.

2 Related Work

There has been considerable research on all of the three sub-tasks of federated search: i.e. i) acquiring resource description, ii) resource selection and iii) results merging. Since this paper focuses on the resource selection task, we mainly survey related work in resource selection and briefly discuss the works in resource description and results merging.

The STARTS (Gravano et al. 1997) protocol is one of the solutions to acquire the resource descriptions. It requires explicit cooperation from each information source. Although STARTS is a good solution in environments where cooperation can be guaranteed, it doesn't work in multi-party environments where some information sources may not cooperate. Query Based Sampling (QBS) (Callan 2000; Callan and Connell 2001) is an alternative approach for acquiring resource descriptions since it doesn't require explicit cooperation from available sources. QBS only uses the normal process of running queries and retrieving a list of downloadable documents to construct resource descriptions. This solution has been shown to acquire rather accurate resource descriptions using a relatively small number of randomly generated queries to retrieve relatively small number of documents.

There is a large body of prior research on resource selection (e.g., Cetintas et al. 2009; Craswell 2000; French et al. 1999; Fuhr 1999; Gravano et al. 1999; Hawking and Thistlewaite 1999; Ipeirotis and Gravano 2002; Nottelmann and Fuhr 2003; Powell et al. 2000; Shokouhi 2007; Shokouhi and Zobel 2007; Si and Callan 2003a; Si and Callan 2004; Zhai and Lafferty 2001).

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. "Big Document" methods include CORI (Callan 2000; Callan et al. 1995), gGlOSS (Gravano et al. 1997; Gravano et al. 1999) and CVV (Yuwono and Lee 1997).

Particularly, the CORI resource selection algorithm has been shown to be one of the most stable and effective "Big Document" resource selection algorithms in prior studies (e.g, Craswell 2000; French et al. 1999). The CORI resource selection algorithm (Callan 2000; Callan et al. 1995) represents each information source by using the words that it contains, their frequencies, and a small number of corpus statistics. Resource ranking is generated by using a Bayesian inference network and an adaptation of the Okapi term frequency normalization. Details can be found in (Callan 2000; Callan et al. 1995).

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 (Si and Callan 2003a).

ReDDE (Si and Callan 2003a) 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 as resource descriptions. 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 (which has already been generated by QBS) to simulate the property of CCDb. In summary, ReDDE issues a query to an index created over CSDb and scores each resource i) proportional to the number of top-ranked documents originating from that resource and ii) considering the estimated size of that resource. More detailed information about CCDb and CSDb databases can be seen on Figure 1.

In particular, ReDDE utilizes the ranked list of a user query on CSDb to simulate the ranked list of this user query on CCDb. With the ranked list of a user query on CSDb, ReDDE estimates the number of relevant documents to query q in an information source C_j as follows:

$$\hat{Rel}\_q(j) = \sum_{d_i \in C_j\_samp} P(rel \mid d_i) * \frac{1}{N_{C_j\_samp}} * \hat{N}_{C_j} \qquad (1)$$

where $\hat{N}_{C_j}$ is the estimated database size for the information source C_j (e.g., by Sample-Resample method [24]); C_{j_samp} is the set of documents that are sampled from database C_j; and $N_{C_{j\_samp}}$ is the number of documents in C_{j_samp} (i.e. the size of C_{j_samp}). The only unknown in Equation 1 is $P(rel \mid d_i)$, i.e. the probability of relevance given a specific document.

$P(rel \mid d_i)$, the probability of relevance of a specific document in Equation 1, is defined as the probability of relevance given document rank when a CCDb is searched by an effective retrieval method (i.e., $Rank\_central(d_i)$) as follows:

$$P(rel \mid d_i) = \begin{cases} C_q & if\ Rank\_central(d_i) < ratio * \hat{N}_{all} \\ 0 & otherwise \end{cases} \quad (2)$$

where C_q is a query dependent constant, $\hat{N}_{all}$ is the estimated total number of documents in the CCDb and ratio is a parameter which is set to be 0.003 as in prior experiments (Si and Callan 2003a). Although this approximation of relevance probability is rather rough, similar methods have been used by many automatic relevance feedback methods.

Rank_central(d_i), the rank of a document d_i in the CCDb, is calculated as follows:

$$Rank\_central(\ d_i\ ) = \sum_{Rank\_Samp(d_j) < Rank\_Samp(d_i)} \frac{\hat{N}_{c(d_j)}}{N_{c(d_j)\_samp}} \quad (3)$$

where Rank_Samp(d_i) is the corresponding rank in CSDb.

$\hat{Rel}\_q(j)$ can be calculated by plugging Equations 3 and 2 into Equation 1, so that the distribution of relevant documents in different information sources is acquired without a need for relevance judgment. The distribution of relevant documents among information sources is sufficient to rank the available information sources for the resource selection task.

However, all the above 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. The prior research ignored a large amount of valuable information like the results from past queries or query logs. 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 not only motivates the research to make use of the results of past queries, but also makes it possible to use the combined approach of utilizing both the valuable results from past queries and the static information of available information sources for the resource selection decision of a current user query.

Voorhees et al.'s work considered using results of past queries (1995). However, the task they are dealing with is not exactly resource selection since their method retrieves documents from all information sources. For the decision of how many documents to retrieve from each source, the work in (Voorhess et al. 1995) proposed two approaches. One method is to calculate the relevant document distribution among information sources for a query q by averaging the relevant document distributions (among all sources) of k most similar past queries with q. The other method is to compute the weights of each information source's most similar query cluster with the query q and use these weights. Both of these methods require relevance judgment.

Some recent research has been conducted for selecting vertical search engines (e.g., Arguello et al. 2009; Diaz et al. 2009). In particular, research work has been conducted to utilize query log and user feedback information to improve vertical search engine selection. However, the research of selecting vertical engines was different from our proposed work in several perspectives. First, they only studied and evaluated selecting one vertical engine for each user query while there are often quite a few relevant resources for each user query in traditional distributed information retrieval applications (e.g., digital libraries). Second, they treated vertical engine selection as a

classification approach, while our work ranks resources by the number of relevant documents contained in available resources. Third, they utilized human relevance judgment, while the proposed work only utilizes results from past queries that are automatically generated. Finally, their approach is a pure resource selection method, while the proposed method is a more complete solution that includes results merging.

Recently, Cetintas et al. showed that using the search results of past queries improves the effectiveness of resource selection; however, their work was limited. Particularly, they did not integrate the two approaches of learning from past queries and using static sampled information. Furthermore, they did not explore the effect of i) using larger amount of past queries, ii) more similar past queries, iii) selecting more information sources, iv) using different query similarity estimation techniques on resource selection effectiveness (2009).

The last step of distributed information retrieval is merging ranked lists produced by different search engines. It is usually treated as a problem of transforming database-specific document scores into database-independent document scores. The CORI results merging formula (Callan 2000) is a heuristic linear combination of the database score and the document score. The Semi-Supervised Learning (SSL) algorithm (Si and Callan 2003b) uses the documents acquired by query-based sampling as training data and utilizes linear regression to learn merging models. When it is necessary, the Semi-Supervised Learning method downloads a small number of documents on the fly to create training data for building the regression models. Please note that SSL will be used in this word for estimating the relevance of information sources to past queries. The documents downloaded by SSL will be an important data source that will be explained later. More detailed information about the set of SSL-downloaded documents along with CCDb, CSDb databases and how they are used in this work can be seen on Figure 1.

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. Furthermore, we propose a combined resource selection approach that utilizes both the information from past queries and the static information from query-based sampling for more accurate resource selection.

The first subsection presents the qSim algorithm for how to use search results from past queries to guide resource selection for online user queries. The second subsection describes the combined resource selection approach.

3.1 Resource Selection by Learning the Results from Past Queries

Assume that there exists a set of past queries, which is denoted by $P = \{p_1, p_2, \ldots, p_M\}$, where p_i represents the i^{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 (details of the query similarity estimation approaches used in this work can be found in Section 3.1.2), let us denote the similarity by $Sim(p_i \mid q)$. Denote the

set of information sources by $S = \{s_1, s_2, \ldots, s_N\}$. For a query q, we use $Rel(s_j \mid q)$ 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. We are interested in ranking the available information sources by comparing their values of the Rel function. Please note that the absolute values of $Sim(p_i \mid q)$ and $Rel(s_j \mid q)$ are not important. Only the relative difference between $Sim(p_{i1} \mid q)$ and $Sim(p_{i2} \mid q)$ is important, for any pair of p_{i1} and p_{i2} while q is fixed; which is also the case for $Rel(s_j \mid q)$.

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

The algorithmic framework of qSim is as follows:

1. Estimate the relevance of an information source s_j to a past query p_i, i.e. estimate $Rel(s_j \mid p_i)$.

   This can be done in an offline manner for past queries. After the federated search system has processed an online query, this query can also be added to the set of past queries for future reference.
2. Estimate the similarity between a past query p_i and an online/test query q, i.e. estimate $Sim(p_i \mid q)$.
3. Estimate the relevance of an information source s_j to an online/test query q, i.e. estimate $Rel(s_j \mid q)$, according to the estimated relevance information of similar past queries (to this online query q).
4. Rank the information sources according to $Rel(s_j \mid q)$, which will give the most relevant information sources to the online query q.

3.1.1 Estimating the Relevance of Information Sources to Past Queries (i.e. $Rel(s_j \mid p_i)$)

Any resource selection algorithm can be applied to estimate the value of $Rel(s_j \mid p_i)$. If a resource selection algorithm can explicitly give scores for all information sources, we can directly map those scores to $Rel(s_j \mid p_i)$. For example, in the ReDDE algorithm, we can set $Rel(s_j \mid p_i) \sim \hat{Rel\_p}_i(j)$, where $\hat{Rel\_p}_i(j)$ is the estimated number of relevant documents in s_j with respect to p_i by ReDDE.

In this work, a regression-based results merging approach (Si and Callan 2003b) is used to get a much more precise $Rel(s_j \mid p_i)$ estimation by utilizing the search results from past queries, no matter whether a prior resource selection has been made for p_i or not. The key intuition underlying this approach is that the more relevant documents from an information source are in the merged ranked list of past query p_i, the more relevant this information source to p_i is. The process of this approach is as follows:

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 mode, we can have $SEL\_p_i=S$, i.e. all information sources.

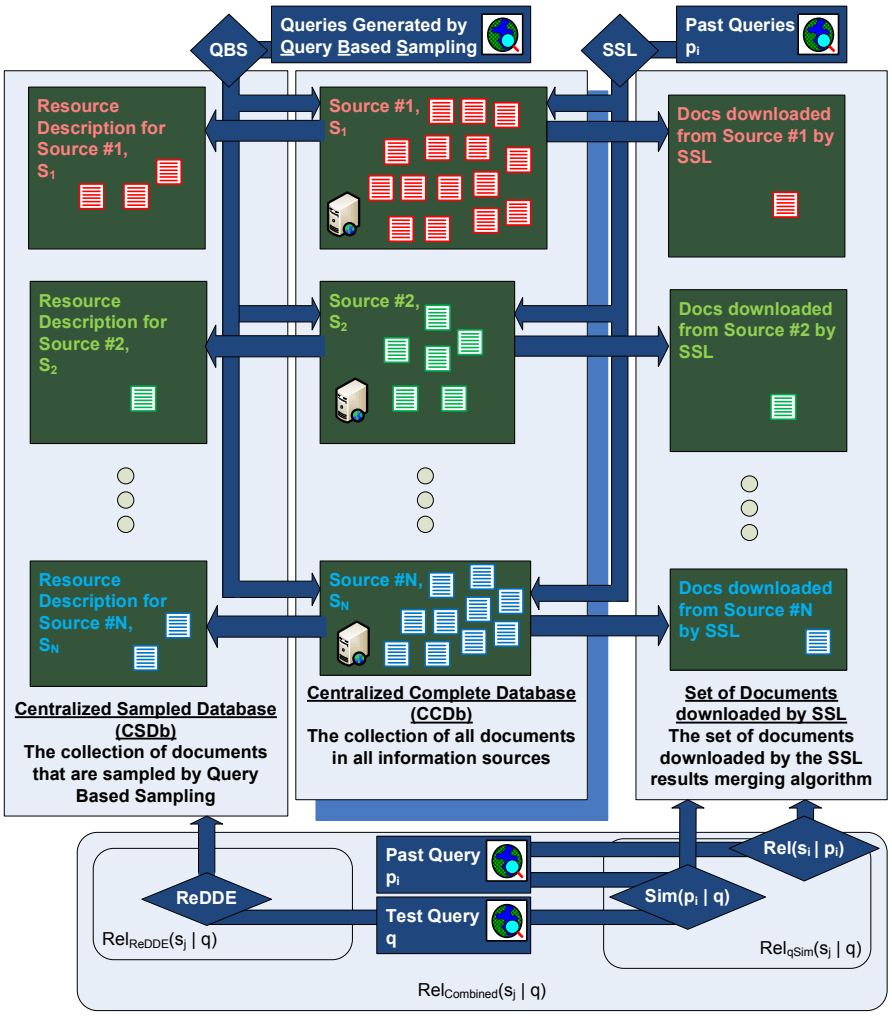


Fig. 1 Sources of evidence for ReDDE and qSim resource selection approaches. A federated search environment consists of several information sources. The set of all documents in all information sources constitute the Centralized Complete Database (CCDd) (shown on left). Note that CCDb is a hypothetical database, which is not available in a federated search environment. Therefore Query Based Sampling (QBS) samples documents from every information source in order to acquire the resource descriptions. The set of all documents sampled by QBS form the centralized sampled database (CSDb) which is used by ReDDE resource selection algorithm (Callan 2000; Callan and Connell 2001). In this work, the combined resource selection approach that uses the resource selection decisions of ReDDE, indirectly uses the CSDb (since ReDDE uses it). For estimating of the relevance of information sources to past queries, SSL results merging algorithm (Si and Callan 2003b) is used. In order to build the regression models, SSL downloads a particular number of documents from each source; and the set of all downloaded document constitute the SSL-downloaded documents database (shown on right). Note that SSL-downloaded documents are used for the estimation of relevance of information sources to past queries (i.e. $Rel(s_j \mid p_i)$) and for retrieval-based query similarity estimation (i.e. $Sim(p_i \mid q)$). Some statistics about the number of documents in each database can be found in the discussion in Section 4.3

Second, we use the regression based Semi-Supervised Learning (SSL) method (Si and Callan 2003b) to merge the search results from those sources in SEL_p_i into a single ranked list. SSL will give us the ranked list of the most relevant documents from all information sources in SEL_p_i to the past query p_i.

Third, the information sources in SEL_p_i are assigned relevance scores proportional to how many of their documents are in the single ranked list of relevant documents to p_i (as; if a source is more relevant to pi, more documents from this source will appear in the ranked list). Particularly, if a 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 Rel(s_j | p_i) is estimated by:

$$\mathrm{Re}l(s_j \mid p_i) \propto \sum_{\substack{top\ T\ docs\ in\ the \\ merged\ ranked\ list}} \frac{\mathrm{Re}l(s_j \mid doc)}{T} \tag{4}$$

where T is a pre-defined number, and

$$\mathrm{Re}l(\ s_j\ \mid\ doc\ ) = \begin{cases} 1, & if\ \ doc \in s_j; \\ 0, & otherwise. \end{cases} \tag{5}$$

The value T, i.e. how many documents are considered in the single ranked list, can be very small like 20. In our experiments, it is set to 20. This helps to minimize the number of interactions with information sources. A typical search engine (such as Google, Yahoo, etc.) only returns 10 or 20 search results or document ids in one page by default.

Please note that the SSL algorithm needs to download some documents for each past query p_i as training data to build the regression models for merging the search results from information sources in SEL_p_i in order to generate the single ranked list for that past query p_i. This is due to the fact although SSL can use the documents in the CSDb as the training documents to build the regression models, it is not allowed to do so in order to evaluate the resource selection approach of learning from past queries in a strict manner. The set of all SSL downloaded training documents for all past queries create the database that is used for i) for the estimation of the relevance of information sources to past queries, and ii) for the estimation of the similarities of past queries and online queries (will be explained in the next section). In this work, 3 top documents are downloaded as the training data for each past query p_i to build the regression models for results merging, in a similar manner as the minimum downloading method of the SSL results merging algorithm (Si and Callan 2003b).

3.1.2 Estimating the Similarity Between Past Queries and Online Queries (i.e. Sim(p_i | q))

There are many ways to measure the similarities between queries (i.e. between past queries and an online query), such as term-based approach (cosine measure, edit distance, latent semantic analysis, etc.), selection-based approach and retrieval-based approach (by comparing the retrieval lists). In our experiments, we use two approaches: i) a retrieval-based approach and ii) term-based approach (cosine similarity) to calculate the similarities. Although the retrieval-based approach is used as the default approach to calculate the query similarities, the results acquired with term-based approach is also used to test the robustness of the proposed algorithms (i.e. to make sure that the proposed techniques work well regardless of the different characteristics of different query similarity estimation approaches).

i) Retrieval-Based Query Similarity Estimation

Retrieval-based query similarity estimation approach compares a past query p_i and an online query q by comparing the similarity of the retrieval results of those queries on a collection. In this work, the set of SSL-downloaded documents (explained in Section 3.1.1.) are used as the collection that queries to be compared are searched on.

Particularly, let R_p_i (or R_q) denote the search results (i.e. the ranked lists) of p_i (or q) on the SSL-downloaded documents database. The similarity of p_i with respect to q is measured by

$$Sim(p_i \mid q) \propto \frac{1}{|R\_p_i|} \sum_{doc \in R\_p_i \cap R\_q} Score(doc, R\_p_i, R\_q) \tag{6a}$$

where the score function for a matching document is given by:

$$Score(doc, R\_p_i, R\_q) = 1 - \left| \frac{doc\ rank\ in\ R\_p_i}{|R\_p_i|} - \frac{doc\ rank\ in\ R\_q)}{|R\_q|} \right| \tag{6b}$$

The value of $Sim(p_i \mid q)$ will be higher if the set of documents that are common to R_p_i and R_q rank similarly in R_p_i and R_q. 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; and normalize the values of $Sim(p_i \mid q)$ in a simple approach as follows:

$$\begin{aligned} MAXSIM\_q &= \max_i Sim(p_i \mid q) \\ SIMTHRES\_q &= 0.8 \times MAXSIM\_q \end{aligned} \tag{7}$$

$$normalized\ Sim(p_i \mid q) \propto \begin{cases} 0, & if\ Sim(p_i \mid q) < SIMTHRES\_q; \\ \dfrac{(Sim(p_i \mid q) - SIMTHRES\_q)}{(MAXSIM\_q - SIMTHRES\_q)}, & otherwise. \end{cases} \tag{8}$$

Retrieval-based similarity estimation approach is used as the default query similarity estimation approach in this work due to the fact that term-based similarity estimation approaches are not as successful in finding the similarity between past and online queries when queries have limited number of terms. More detailed discussion on the comparison of retrieval-based similarity approach and term-based similarity approach can be seen on Figure 2.

ii) Term-Based Query Similarity Estimation (Cosine Measure)

As a second method of calculating the similarities between past queries and online queries, we use the common cosine similarity (Baeza-Yates and Ribeiro-Neto 1999). If we denote the bag-of-words vector representations of past query p_i and online query q as $\overrightarrow{p_i}$ and $\vec{q}$, the cosine similarity is given by:

$$Sim(p_i \mid q) \propto \cos(\overrightarrow{p_i}, \vec{q}) = \frac{\overrightarrow{p_i} \bullet \vec{q}}{\left\|\overrightarrow{p_i}\right\| * \left\|\vec{q}\right\|} \tag{9}$$

where "$\bullet$" is the dot product of these vectors. We use the common Okapi weighting scheme to calculate similarities between past queries and online queries; and do normalization to scale the similarity scores between 0 and 1.

We use term-based similarity as the secondary method of query similarity estimation approach and used it only with the retrieval-based query similarity approach to test the robustness of the proposed algorithms (i.e. to make sure that the proposed techniques work well regardless of the different characteristics of different query similarity estimation approaches). More detailed discussion on the comparison of retrieval-based similarity approach and term-based similarity approach can be seen on Figure 2.

3.1.3 Estimating the Relevance of Information Sources to Online Queries (i.e. Rel(s_j | q))

For an online query q, we predict the relevance of information sources (i.e. Rel(s_j | q)) based on the estimated relevance information from similar past queries. It is calculated by:

$$\text{Rel}(s_j \mid q) \propto \sum_i \text{Rel}(s_j \mid p_i)\,\text{Sim}(p_i \mid q) \tag{10}$$

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.

3.1.4 Resource Selection According to Rel(s_j | 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.

As we mentioned before, we can add a query q into the set of past queries after this query has been processed.

3.1.5 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.

3.2 Combined Approach for Resource Selection

In this subsection, we propose a simple way to combine qSim with other resource selection algorithms like ReDDE to achieve better performance.

In particular, we focus on the combination of qSim and ReDDE. Assume that ReDDE generates the score of a particular information source s_j as $\text{Rel}\_{\text{ReDDE}}(s_j \mid q)$ for resource selection. We denote the corresponding resource selection score from qSim as $\text{Rel}\_{\text{qSim}}(s_j \mid q)$, and then we can simply combine the values of $\text{Rel}\_{\text{ReDDE}}(s_j \mid q)$ and $\text{Rel}\_{\text{qSim}}(s_j \mid q)$ in a linear way:

$$\mathrm{Rel\_combined}(s_j \mid q) \propto \lambda \frac{\mathrm{Rel\_qSim}(s_j \mid q)}{\max_j \mathrm{Rel\_qsim}(s_j \mid q)} + (1-\lambda)\frac{\mathrm{Rel\_ReDDE}(s_j \mid q)}{\max_j \mathrm{Rel\_ReDDE}(s_j \mid q)} \quad (11)$$

where λ is a real number within [0,1]. In our experiments, λ is set to 1/3. Finally, we generate the resource selection results according to the values of $\mathrm{Rel\_combined}(s_j \mid q)$.

4 Experimental Methodology

It is most desirable to evaluate federated search algorithms with real world applications. However, real world applications are usually short of relevance judgment data and it is difficult to obtain full control of different components of the systems (e.g., varying retrieval algorithms of each information source). These difficulties prevent us from doing thorough study with real world applications. Instead, an extensive set of experiments was designed in research environments to simulate real world applications such as (Avrahami et al. 2006).

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 testbed, 50 queries were created from the title fields of TREC topics 51-100. For Trec4, another 50 queries were created from the description fields of TREC topics 201-250. These queries will be used as the set of test/online queries and will be referred to test, online or real queries throughout the paper. The detailed statistics about the test queries can be found in Table 2.

All the 100 information sources were assigned one of three types of retrieval algorithms as INQUERY (Callan et al. 1995), a unigram language model with linear smoothing (Lemur Toolkit; Zhai and Lafferty 2001) and a TFIDF retrieval algorithm with the "ltc" weighing schema (Si and Callan 2003b). All the algorithms were implemented with the lemur toolkit (Lemur Toolkit, Ogilvie and Callan 2001).

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. In order to better simulate the characteristics of a real world federated search system, 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 datasets are used as the set of test/online queries and either one of the two sets of generated past queries are

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

Table 2. TREC query set statistics.

Collections	TREC Topic Set	TREC Topic Field	Average Length (Words)
Trec123	51-100	Title	2.92
Trec4	201-250	Description	8.82

used as the set of past queries. Specifically the experiments are conducted separately on the two sets of generated past queries to test the robustness of the proposed algorithms (i.e. to make sure that the proposed techniques work well regardless of the different characteristics of the sets of generated 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*. 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.* To reiterate; the experiments are using either the sampled past queries or the simulated past queries as the set of past queries. More detailed explanation of these two approaches is given in the following two subsections.

4.2.1 Sampled Past Queries

This subsection describes the first approach to generate the sampled past queries. Two sets of sampled past queries are generated for Trec123, which are called Trec123_T20 and Trec123_T10. The past queries in Trec123_T20 (or Trec123_T10) 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. The documents in CSDB of ReDDE are particularly excluded in order not to exploit the advantage of the documents of CSDB of ReDDE that are used as resource descriptors.
- For each test query, TOPX 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 X sampled queries for this test query (i.e. each title becomes a new sampled past query). Please note that X is 10 for Trec123_T10 and 20 for Trec123_T20 query set.

For 50 test queries, the above sampling process is repeated and a total of 1000 past queries are acquired for Trec123_T20 and a total of 500 past queries are acquired for Trec123_T10. In the

Table 3a. Statistics of sampled past queries.

Collections	Sampled Past Query Set	# of TOP Docs (titles extracted)	Average Length (Words)
Trec123	Trec123_T20	20	8.98
Trec123	Trec123_T10	10	8.80
Trec4	Trec4_T20	20	8.53
Trec4	Trec4_T10	10	8.42

Table 3b. Statistics of simulated past queries.

Collections	Simulated Past Query Set	# of Words Removed	Average Length (Words)
Trec123	Trec123_R1	1	2.60
Trec123	Trec123_R2	2	2.16
Trec4	Trec4_R2	2	6.92
Trec4	Trec4_R3	3	6.12

same way described above, two sets of sampled past queries are also generated for Trec4 testbed, which are called Trec4_T20 and Trec4_T10. Trec4_T20 & Trec4_T10 have 1000 & 500 sampled queries respectively.

In the experiments that used the sampled past queries as the set of past queries; the 50 real queries are treated as test/online user queries, and the 1000 sampled past queries (i.e. the past queries in Trec123_T20 or Trec4_T20 sampled past query sets) or 500 sampled queries (i.e. in Trec123_T10 or Trec4_T10 sampled past query sets) are treated as the past queries. Details about sampled past queries can be seen in Table 3a.

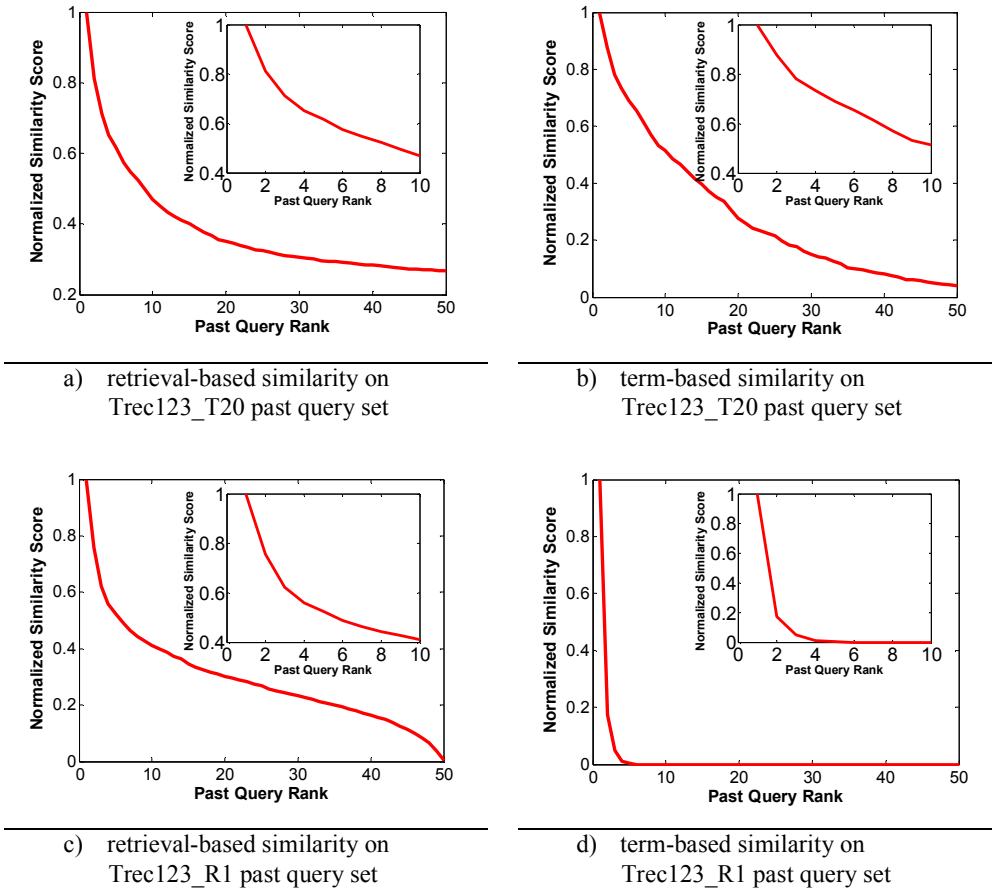
4.2.2 Simulated Past Queries

This subsection describes the second approach to generate the simulated past queries. Two sets of simulated past queries are generated for Trec123, which are called Trec123_R1 and Trec123_R2. Each set contains 50 simulated queries. The past queries in Trec123_R1 (or Trec123_R2) were generated in the following way:

- For each test query, 1 (or 2) term(s) is (are) removed to generate a simulated past query;
- For each simulated past query at least 2 terms are kept to make sure the simulated past query is not too short to be meaningful. (e.g., in Trec123_R2, if a real query only contains 3 terms, only 1 term is removed instead of 2 to make sure the simulated query contains at least 2 terms.)

For Trec4 testbed, two levels of simulated past queries are also generated, which are called Trec4_R2 and Trec4_R3. Each set contains 50 simulated queries. The past queries in Trec4_R2 (or Trec4_R3) were generated by the following way:

- For each test query, 2 (or 3) terms are randomly removed to generate a simulated past query;
- For each simulated past query at least 3 terms are kept.



a) retrieval-based similarity on Trec123_T20 past query set

b) term-based similarity on Trec123_T20 past query set

c) retrieval-based similarity on Trec123_R1 past query set

d) term-based similarity on Trec123_R1 past query set

Fig. 2 Statistics of Retrieval-Based and Term-Based Similarity measures over Trec123_T20 and Trec123_R1 past query sets. Each online/test query has a set of similar past queries sorted with respect to the normalized similarity scores while determining the set of most similar past queries. The distribution of the normalized similarity scores between an online query and the set of past queries give the information of the normalized similarity score of the most similar past query at a particular rank with respect to an online query (i.e. the normalized similarity score of the k^{th} most similar past query with the online query q). The average of the distribution of normalized similarity scores across all online queries (y-axes) for ranks 1-to-50 (x-axes) is calculated for Trec123_T20 and Trec123_R1 past query datasets with retrieval-based and term-based similarity measures and is reported in each graph. Detailed view of each graph for the top 10 most similar past queries (note that in this work we only choose the top 5 most similar past queries for each test query) can be seen on the smaller graphs on the right top of each graph.

An important observation is that for queries that have very limited number of terms (such as Trec123_R1 and Trec123_R2) term-based similarity measure cannot calculate the similarity between the past and online queries as precise as the retrieval based similarity approach as shown on Graphs c & d. This is due to the fact that although there is some similarity between a past query and an online query, term-based approach cannot detect it if there are no common terms. But when the queries have enough number of terms, term-based approach also performs as good as the retrieval-based approach in finding the similarity between past and online queries as shown on Graphs a & b.

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, Trec123_R2, Trec4_R2 or Trec4_R3 simulated past query sets) are treated as past queries. Generally speaking, simulated past queries in Trec123_R1 are more similar than Trec123_R2 with

respect to the Trec123 online queries and simulated past queries in Trec4_R2 are more similar than Trec4_R3 with respect to the Trec4 online queries. Details about simulated past queries can be seen in Table 3b.

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.

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 qSim, there are either 1000 or 500 past queries at most (i.e. for sampled queries), and for each past query 3 top documents per source are downloaded by SSL in order to build a regression model to merge search results into a single ranked list. So in total, we need to download at most 3000 or 1500 documents per source. The number is actually much smaller than 3000 or 1500 as only a very small portion of sources are selected for searching each past query: for *qSim-Cut-10* on Trec123(or Trec4)_T10 we download about 150 and for *qSim-Cut-10* on Trec123(or Trec4)_T20 we download about 300 documents per source on average.

For ReDDE, we build the CSDb, which contains 150 documents per source by QBS. All other parameters in ReDDE are the same as that in [23].

Although the types of evidence that qSim and ReDDE use are different and it will not be perfectly fair to make a comparison, they are still comparable by the amount of documents they use.

4.4 Evaluation Metric

The recall metric R_k has been commonly used to evaluate resource selection algorithms [2,7,23]. Let B denote a desirable ranking of available information sources by Relevance-Based Ranking (i.e., ranking of sources by the actual number of relevant documents), and E a ranking provided by a resource selection algorithm. Let B_i and E_i denote the number of relevant documents in the i^{th} ranked database of B or E. The metric R_k is defined as follows:

$$R_k = \frac{\sum_{i=1}^{k} E_i}{\sum_{i=1}^{k} B_i} \tag{11}$$

The metric above 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

An extensive set of experiments are conducted to address the following six questions:

Table 4a. Comparison between ReDDE with qSim-Cut-10 and qSim-Cut-20 on Trec123 dataset with Trec123_T20 sampled past query set. Please note that the results for qSim-Cut-10 on Trec123 dataset with Trec123_T20 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec123	qSim-Cut-10 on Trec123_T20	qSim-Cut-20 on Trec123_T20
1	0.2445	0.3414 (39.66%)	0.3961 (62.02%)
2	0.3121	0.3675 (17.73%)	0.4317 (38.30%)
3	0.3188	0.4021 (26.15%)	0.4108 (28.86%)
4	0.3350	0.3929 (17.31%)	0.4326 (29.17%)
5	0.3503	0.4108 (17.26%)	0.4332 (23.67%)
6	0.3745	0.4093 (09.30%)	0.4370 (16.70%)
7	0.3889	0.4194 (07.84%)	0.4475 (15.08%)
8	0.4025	0.4373 (08.66%)	0.4686 (16.44%)
9	0.4108	0.4429 (07.83%)	0.4865 (18.45%)
10	0.4325	0.4577 (05.83%)	0.5101 (17.94%)
Overall Improvement		15.76%	26.67%

Table 4b. Comparison between ReDDE with qSim-Cut-10 and qSim-Cut-20 on Trec4 dataset with Trec4_T20 sampled past query set. Please note that the results for qSim-Cut-10 on Trec4 dataset with Trec4_T20 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec4	qSim-Cut-10 on Trec4_T20	qSim-Cut-20 on Trec4_T20
1	0.2739	0.3221 (17.65%)	0.2514 (-08.16%)
2	0.3375	0.3288 (-02.56%)	0.3675 (08.89%)
3	0.3031	0.3295 (08.73%)	0.3642 (20.17%)
4	0.3101	0.3499 (12.85%)	0.3551 (14.51%)
5	0.3095	0.3590 (16.01%)	0.3606 (16.51%)
6	0.3247	0.3623 (11.60%)	0.3746 (15.39%)
7	0.3339	0.3654 (09.44%)	0.3842 (15.08%)
8	0.3599	0.3681 (02.29%)	0.3974 (10.42%)
9	0.3753	0.3678 (-01.96%)	0.4010 (06.87%)
10	0.3905	0.3767 (-03.52%)	0.4077 (04.41%)
Overall Improvement		7.05%	10.41%

1. *How good is the new resource selection algorithm by learning results from past queries?* Experiments are conducted to compare the new resource selection algorithm with the state-of-the-art ReDDE resource selection that uses static sampled information for resource selection.
2. *How does the new resource selection algorithm behave when different amount of information sources have been searched for past queries?* Experiments are conducted to show the performance of the new resource selection algorithm when different amount of information sources have been searched for past queries.
3. *How does the new resource selection algorithm behave with different amount of past queries?* Experiments are conducted to show the performance of the new resource selection algorithm with more or less amount of past queries.
4. *How does the new resource selection algorithm behave with different characteristics of past queries?* Experiments are conducted to show the performance of the new resource selection algorithm with more or less similar past queries.

Table 5a. Comparison between ReDDE with qSim-Cut-10 and qSim-Cut-20 on Trec123 dataset with Trec123_R1 simulated past query set. Please note that the results for qSim-Cut-10 on Trec123 dataset with Trec123_R1 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec123	qSim-Cut-10 on Trec123_R1	qSim-Cut-20 on Trec123_R1
1	0.2445	0.4003 (63.71%)	0.4126 (68.78%)
2	0.3121	0.4229 (35.47%)	0.4523 (44.91%)
3	0.3188	0.4242 (33.06%)	0.4653 (45.94%)
4	0.3350	0.4372 (30.53%)	0.4784 (42.81%)
5	0.3503	0.4410 (25.90%)	0.4812 (37.35%)
6	0.3745	0.4336 (15.79%)	0.4790 (27.91%)
7	0.3889	0.4356 (12.02%)	0.4830 (24.21%)
8	0.4025	0.4392 (09.15%)	0.4969 (23.47%)
9	0.4108	0.4454 (08.43%)	0.4995 (21.62%)
10	0.4325	0.4461 (03.14%)	0.5062 (17.06%)
Overall Improvement		23.72%	35.41%

Table 5b. Comparison between ReDDE with qSim-Cut-10 and qSim-Cut-20 on Trec4 dataset with Trec4_R2 simulated past query set. Please note that the results for qSim-Cut-10 on Trec4 dataset with Trec4_R2 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec4	qSim-Cut-10 on Trec4_R2	qSim-Cut-20 on Trec4_R2
1	0.2739	0.2882 (05.23%)	0.3096 (13.07%)
2	0.3375	0.3065 (-09.20%)	0.3294 (-02.41%)
3	0.3031	0.3176 (04.80%)	0.3433 (13.28%)
4	0.3101	0.3306 (06.62%)	0.3473 (11.98%)
5	0.3095	0.3447 (11.37%)	0.3627 (17.19%)
6	0.3247	0.3554 (09.45%)	0.3707 (14.17%)
7	0.3339	0.3665 (09.76%)	0.3794 (13.62%)
8	0.3599	0.3788 (05.24%)	0.3923 (09.00%)
9	0.3753	0.3896 (03.83%)	0.4008 (06.82%)
10	0.3905	0.3972 (01.74%)	0.4159 (06.51%)
Overall Improvement		4.88%	10.32%

5. *How does the new resource selection algorithm behave when different query similarity estimation approaches are used to calculate the similarity between a test query and the set of past queries?* Experiments are conducted to show the performance of the new resource selection algorithm with retrieval-based query similarity estimation approach and term-based query similarity estimation approach.
6. *Can the combined resource selection approach further improve the accuracy of resource selection?* Experiments are conducted to compare the combined approach with the two approaches of either learning from past queries or using static sampled information.

5.1 Comparison between qSim and ReDDE

Table 4a and Table 4b show the results of qSim and ReDDE on the Trec123 and Trec4 testbeds with Trec123_T20 and Trec4_T20 sampled query sets. In the same way Table 5a and Table 5b show the corresponding results with Trec123_R1 and Trec4_R2 simulated query sets.

Table 6a. Comparison between different amounts of sampled past query sets on Trec123 dataset with Trec123_T10 and Trec123_T20 sampled past query sets. Please note that the results for qSim-Cut-10 on Trec123 dataset with Trec123_T20 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec123	qSim-Cut-10 on Trec123_T10	qSim-Cut-10 on Trec123_T20
1	0.2445	0.2702 (10.54%)	0.3414 (39.66%)
2	0.3121	0.3445 (10.37%)	0.3675 (17.73%)
3	0.3188	0.3529 (10.70%)	0.4021 (26.15%)
4	0.3350	0.3701 (10.49%)	0.3929 (17.31%)
5	0.3503	0.3901 (11.36%)	0.4108 (17.26%)
6	0.3745	0.3916 (04.58%)	0.4093 (09.30%)
7	0.3889	0.4062 (04.47%)	0.4194 (07.84%)
8	0.4025	0.4184 (03.97%)	0.4373 (08.66%)
9	0.4108	0.4193 (02.08%)	0.4429 (07.83%)
10	0.4325	0.4334 (00.22%)	0.4577 (05.83%)
Overall Improvement		6.88%	15.76%

Table 6b. Comparison between different amounts of sampled past query sets on Trec4 dataset with Trec4_T10 and Trec4_T20 sampled past query sets. Please note that the results for qSim-Cut-10 on Trec4 dataset with Trec4_T20 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec4	qSim-Cut-10 on Trec4_T10	qSim-Cut-10 on Trec4_T20
1	0.2739	0.3329 (21.56%)	0.3221 (17.65%)
2	0.3375	0.3222 (04.52%)	0.3288 (-02.56%)
3	0.3031	0.3366 (11.07%)	0.3295 (08.73%)
4	0.3101	0.3345 (07.89%)	0.3499 (12.85%)
5	0.3095	0.3217 (03.96%)	0.3590 (16.01%)
6	0.3247	0.3363 (03.57%)	0.3623 (11.60%)
7	0.3339	0.3455 (03.49%)	0.3654 (09.44%)
8	0.3599	0.3448 (-04.20%)	0.3681 (20.29%)
9	0.3753	0.3441 (-08.29%)	0.3678 (-01.96%)
10	0.3905	0.3592 (-07.98%)	0.3767 (-03.52%)
Overall Improvement		2.65%	7.05%

The performance is evaluated by the Recall metric R_k defined in section 4.4, where "# Selected Sources" in the table 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 4a, Table 4b, Table 5a and Table 5b, qSim-Cut-10 always generates much better results than ReDDE. Thus, qSim can be considered as a very effective algorithm for resource selection.

5.2 The Performance of qSim with Different Amounts of Search Results from Past Queries

Table 4a & Table 4b (with sampled past queries) and Table 5a & Table 5b (with simulated past queries) also show that searching more information sources (i.e. X = 20 in this case) for past queries can help us do a better resource selection for online queries (than searching fewer information sources (i.e. X=10)). This is a reasonable result as the search results of past queries in qSim-Cut-20 provide more information than the search results of past queries in qSim-Cut-10.

Table 7a. Comparison between different similarity levels of simulated past query sets on Trec123 dataset with Trec123_R2 and Trec123_R1 simulated past query sets. Please note that the results for qSim-Cut-10 on Trec123 dataset with Trec123_R1 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec123	qSim-Cut-10 on Trec123_R2	qSim-Cut-10 on Trec123_R1
1	0.2445	0.3745 (53.16%)	0.4003 (63.71%)
2	0.3121	0.3887 (24.53%)	0.4229 (35.47%)
3	0.3188	0.3923 (23.03%)	0.4242 (33.06%)
4	0.3350	0.3996 (19.31%)	0.4372 (30.53%)
5	0.3503	0.4042 (15.38%)	0.4410 (25.90%)
6	0.3745	0.4053 (08.25%)	0.4336 (15.79%)
7	0.3889	0.4106 (05.58%)	0.4356 (12.02%)
8	0.4025	0.4228 (05.07%)	0.4392 (09.15%)
9	0.4108	0.4294 (04.56%)	0.4454 (08.43%)
10	0.4325	0.4309 (-00.35%)	0.4461 (03.14%)
Overall Improvement		15.85%	23.72%

Table 7b. Comparison between different similarity levels of simulated past query sets on Trec4 dataset with Trec4_R3 and Trec4_R2 simulated past query sets. Please note that the results for qSim-Cut-10 on Trec4 dataset with Trec4_R2 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec4	qSim-Cut-10 on Trec4_R3	qSim-Cut-10 on Trec4_R2
1	0.2739	0.2828 (03.27%)	0.2882 (05.23%)
2	0.3375	0.3065 (-09.20%)	0.3065 (-09.20%)
3	0.3031	0.3183 (05.04%)	0.3176 (04.80%)
4	0.3101	0.3237 (04.38%)	0.3306 (06.62%)
5	0.3095	0.3340 (07.92%)	0.3447 (11.37%)
6	0.3247	0.3424 (05.44%)	0.3554 (09.45%)
7	0.3339	0.3567 (06.84%)	0.3665 (09.76%)
8	0.3599	0.3727 (03.55%)	0.3788 (05.24%)
9	0.3753	0.3823 (01.86%)	0.3896 (03.83%)
10	0.3905	0.3901 (-00.09%)	0.3972 (01.74%)
Overall Improvement		2.90%	4.88%

However, we should be aware that searching more information sources will increase the costs, i.e. it will require more documents to be downloaded and lead to more computational costs for merging the search results. In a federated search environment of 100 information sources, qSim-Cut-10 may be a more practical algorithm for resource selection.

5.3 The Performance of qSim with Different Amounts of Past Queries

The results in Table 6a and Table 6b indicate that the performance of qSim-Cut-10 is better with Trec123_T20 and Trec4_T20 past query sets than Trec123_T10 and Trec4_T10 past query sets respectively. This demonstrates that more past queries can provide better information for qSim to achieve better resource selection recalls. In a real world scenario, user queries may be similar or even duplicates. Therefore, a search system can effectively utilize the search results of more amounts of past queries to get better performance.

Table 8a. Comparison between qSim-Cut-10 with retrieval-based similarity and qSim-Cut-10 with term-based similarity on Trec123 dataset with Trec123_T20 sampled past query set. Please note that the results for qSim-Cut-10 on Trec123 dataset with Trec123_T20 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec123	qSim-Cut-10 with Retrieval-Based Similarity on Trec123_T20	qSim-Cut-10 with Term-Based Similarity on Trec123_T20
1	0.2445	0.3414 (39.66%)	0.3951 (61.60%)
2	0.3121	0.3675 (17.73%)	0.3982 (27.55%)
3	0.3188	0.4021 (26.15%)	0.4558 (42.95%)
4	0.3350	0.3929 (17.31%)	0.4537 (35.47%)
5	0.3503	0.4108 (17.26%)	0.4782 (36.50%)
6	0.3745	0.4093 (09.30%)	0.4798 (28.12%)
7	0.3889	0.4194 (07.84%)	0.4883 (25.57%)
8	0.4025	0.4373 (08.66%)	0.4989 (23.97%)
9	0.4108	0.4429 (07.83%)	0.5061 (23.21%)
10	0.4325	0.4577 (05.83%)	0.5030 (16.31%)
Overall Improvement		15.76%	32.12%

Table 8b. Comparison between qSim-Cut-10 with retrieval-based similarity and qSim-Cut-10 with term-based similarity on Trec4 dataset with Trec4_T20 sampled past query set. Please note that the results for qSim-Cut-10 on Trec4 dataset with Trec4_T20 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec4	qSim-Cut-10 with Retrieval-Based Similarity on Trec4_T20	qSim-Cut-10 with Term-Based Similarity on Trec4_T20
1	0.2739	0.3221 (17.65%)	0.3437 (25.49%)
2	0.3375	0.3288 (-02.56%)	0.4067 (20.51%)
3	0.3031	0.3295 (08.73%)	0.3955 (30.50%)
4	0.3101	0.3499 (12.85%)	0.3750 (20.93%)
5	0.3095	0.3590 (16.01%)	0.3713 (19.97%)
6	0.3247	0.3623 (11.60%)	0.3802 (17.10%)
7	0.3339	0.3654 (09.44%)	0.4031 (20.72%)
8	0.3599	0.3681 (02.29%)	0.4114 (14.29%)
9	0.3753	0.3678 (-01.96%)	0.4152 (10.64%)
10	0.3905	0.3767 (-03.52%)	0.4211 (07.85%)
Overall Improvement		7.05%	18.80%

5.4 The Performance of qSim with Different Similarity Levels of Past Queries

The results in Table 7a and Table 7b indicate that the performance of qSim-Cut-10 is better with Trec123_R1 and Trec4_R2 past query sets than Trec123_R2 and Trec4_R3 past query sets respectively. This demonstrates that more similar past queries with respect to online/test queries can provide better information for qSim to achieve better resource selection recalls. A search system can effectively utilize the search results of more similar past queries to get better performance.

5.5 The Performance of qSim with Different Query Similarity Estimation Approaches

Table 8a and Table 8b show the results of qSim and ReDDE on the Trec123 testbed with Trec123_T20 sampled query sets. As shown from Table 8a and Table 8b, qSim-Cut-10 always generates much better results than ReDDE when either of the two query similarity estimation

Table 9a. Comparison between qSim-Cut-10 with the Combined Approach on Trec123 dataset with Trec123_T20 sampled past query set. Please note that the results for qSim-Cut-10 on Trec123 dataset with Trec123_T20 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec123	qSim-Cut-10 on Trec123_T20	Combined Approach on Trec123_T20
1	0.2445	0.3414 (39.66%)	0.3580 (46.41%)
2	0.3121	0.3675 (17.73%)	0.3758 (20.38%)
3	0.3188	0.4021 (26.15%)	0.3884 (21.82%)
4	0.3350	0.3929 (17.31%)	0.4035 (20.46%)
5	0.3503	0.4108 (17.26%)	0.4213 (16.21%)
6	0.3745	0.4093 (09.30%)	0.4352 (16.21%)
7	0.3889	0.4194 (07.84%)	0.4432 (13.98%)
8	0.4025	0.4373 (08.66%)	0.4467 (11.00%)
9	0.4108	0.4429 (07.83%)	0.4448 (08.28%)
10	0.4325	0.4577 (05.83%)	0.4567 (05.61%)
Overall Improvement		15.76%	18.44%

Table 9b. Comparison between qSim-Cut-10 with the Combined Approach on Trec4 dataset with Trec4_T20 sampled past query set. Please note that the results for qSim-Cut-10 on Trec4 dataset with Trec4_T20 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec4	qSim-Cut-10 on Trec4_T20	Combined Approach on Trec4_T20
1	0.2739	0.3221 (17.65%)	0.3347 (22.22%)
2	0.3375	0.3288 (-02.56%)	0.3370 (-00.15%)
3	0.3031	0.3295 (08.73%)	0.3374 (11.32%)
4	0.3101	0.3499 (12.85%)	0.3397 (09.54%)
5	0.3095	0.3590 (16.01%)	0.3627 (17.19%)
6	0.3247	0.3623 (11.60%)	0.3607 (11.10%)
7	0.3339	0.3654 (09.44%)	0.3737 (11.91%)
8	0.3599	0.3681 (02.29%)	0.3751 (04.20%)
9	0.3753	0.3678 (-01.96%)	0.3848 (02.55%)
10	0.3905	0.3767 (-03.52%)	0.3949 (01.16%)
Overall Improvement		7.05%	9.10%

approaches, namely retrieval-based query similarity estimation and term-based query similarity estimation, is used. Thus, qSim can be considered as a very effective algorithm for resource selection regardless of the query similarity estimation approach. In this set of experiments, only Trec123_T20 dataset is used. This is because of the fact that both of retrieval-based query similarity estimation and term-based similarity estimation approaches can find the similarities of past queries and online queries successfully on this past query dataset as shown and discussed on Figure 2.

An interesting observation with this set of experiments is that qSim with term-based query similarity estimation approach significantly outperforms qSim with retrieval-based similarity estimation approach. Although it is an interesting direction to explore the effect of using different query similarity estimation approaches on the resource selection effectiveness, it is beyond the scope of this paper. In this paper, we utilize retrieval-based similarity measurement in all of the experiments since this measurement is available for different types of queries and testbeds. As mentioned before, in this set of experiments we are only interested in the robustness of qSim algorithm with respect to different query similarity estimation approaches.

Table 10a. Comparison between qSim-Cut-10 with the Combined Approach on Trec123 dataset with Trec123_R1 simulated past query set. Please note that the results for qSim-Cut-10 on Trec123 dataset with Trec123_R1 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec123	qSim-Cut-10 on Trec123_R1	Combined Approach on Trec123_R1
1	0.2445	0.4003 (63.71%)	0.3982 (62.87%)
2	0.3121	0.4229 (35.47%)	0.4205 (34.72%)
3	0.3188	0.4242 (33.06%)	0.4355 (36.59%)
4	0.3350	0.4372 (30.53%)	0.4407 (31.58%)
5	0.3503	0.4410 (25.90%)	0.4513 (28.80%)
6	0.3745	0.4336 (15.79%)	0.4442 (18.60%)
7	0.3889	0.4356 (12.02%)	0.4390 (12.88%)
8	0.4025	0.4392 (09.15%)	0.4472 (11.11%)
9	0.4108	0.4454 (08.43%)	0.4560 (11.01%)
10	0.4325	0.4461 (03.14%)	0.4543 (05.04%)
Overall Improvement		23.72%	25.32%

Table 10b. Comparison between qSim-Cut-10 with the Combined Approach on Trec4 dataset with Trec4_R2 simulated past query set. Please note that the results for qSim-Cut-10 on Trec4 dataset with Trec4_R2 sampled past query set has been reported in our previous work (Cetintas et al. 2009).

# Selected Sources	ReDDE on Trec4	qSim-Cut-10 on Trec4_R2	Combined Approach on Trec4_R2
1	0.2739	0.2882 (05.23%)	0.2953 (07.84%)
2	0.3375	0.3065 (-09.20%)	0.3126 (-07.39%)
3	0.3031	0.3176 (04.80%)	0.3169 (04.55%)
4	0.3101	0.3306 (06.62%)	0.3288 (06.04%)
5	0.3095	0.3447 (11.37%)	0.3465 (11.96%)
6	0.3247	0.3554 (09.45%)	0.3626 (11.67%)
7	0.3339	0.3665 (09.76%)	0.3713 (11.22%)
8	0.3599	0.3788 (05.24%)	0.3813 (05.95%)
9	0.3753	0.3896 (03.83%)	0.3961 (05.54%)
10	0.3905	0.3972 (01.74%)	0.4094 (04.86%)
Overall Improvement		4.88%	6.22%

5.6 The Performance of Combining qSim and ReDDE

Table 9a & Table 9b (with sampled past queries) and Table 10a & Table 10b (with simulated past queries) compare the performances of ReDDE, qSim and the combined approach. The results show that the combined approach further improves the performance of qSim. The improvement is due to the integration of the two types of approaches: learning from past queries and using static sampled information. It has its advantages to integrate two types of evidence. Especially when a new rare test/online query is encountered, the combined approach will be a good choice for the system to perform at least as good as the ReDDE algorithm or even better than that. Since a real world federated search system usually has access to both types of evidence, the combined approach is a practical solution.

6 Conclusion

Resource selection is an important component for a federated text search system. A large body of resource selection algorithms has been proposed in prior research. Most prior research of resource selection utilized static information of available information sources that is sampled in the offline manner and estimated the utilities of information sources.

However, most prior research ignored a large amount of information of the results from past queries. In a real world federated search system, there often exist many similar queries like the case of Web search. The results of similar past queries may provide very valuable information to guide the resource selection decision of a current user query.

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 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. Furthermore, a combined resource selection approach is proposed in this paper to combine the resource selection results of the qSim algorithm and the ReDDE algorithm that uses static information.

An extensive set of experiments are conducted on two testbeds with several configurations to show the effectiveness of the new resource selection algorithms. The proposed qSim resource selection algorithm is compared with the ReDDE algorithm. When there exist similar past queries for test queries, our new qSim algorithm performs better than the ReDDE algorithm. Furthermore when there are new online queries for which there are no similar past queries, the qSim algorithm mostly utilizes the combined approach and still generates substantial improvements over the ReDDE algorithm.

There are several directions to extend the research in this paper. For example, we have shown that different query similarity estimation approaches have different performance gains; therefore a more detailed analysis of several query similarity estimation approaches and utilization of a more sophisticated query similarity estimation approach may help better identify similar past queries for more accurate resource selection results. Another possibility is that we can propose query-specific combination approach, which automatically adjusts the weights on the ranked lists from qSim and ReDDE with respect to the characteristics of user queries.

7 References

Avrahami , T. T., Yau, L., Si, L., and Callan, J. (2006). The FedLemur project: Federated search in the real world. Journal of the American Society for Information Science and Technology.

Arguello, J., Diaz, F., Callan, J., and Crespo, J. (2009). Sources of evidence for vertical selection. In Proceedings of the 32nd international ACM SIGIR Conference on Research and Development in information Retrieval.

Baillie, M., Carman, M. J. and Crestani, F. (2009). A topic-based measure of resource description quality for distributed information retrieval. In Proceedings of the 31st European Conference on Information Retrieval, pp. 485-496.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval.* ACM Press, Series/Addison Wesley, 75-82.

Callan, J. (2000). Distributed information retrieval. In W.B. Croft, editor, *Advances in Information Retrieval*. Kluwer Academic Publishers. (pp. 127-150).

Callan, J. and Connell, M. (2001). Query Based Sampling of Text Databases. In *ACM Transactions on Information Systems*, 19(2). (pp. 97-130).

Callan, J., Croft, W.B., and Broglio, J. (1995). TREC and TIPSTER experiments with INQUERY. Information Processing and Management, 31(3). (pp. 327-343).

Cetintas, S. and Si, L. (2007). Exploration of the tradeoff between effectiveness and efficiency for results merging in federated search. In Proceedings of the 30th Annual international ACM SIGIR Conference on Research and Development in information Retrieval.

Cetintas, S., Si, L., and Yuan, H. (2009). Learning from past queries for resource selection. In Proceedings of the 18th International Conference on Information and Knowledge Management. (to appear in 4 pages).

Craswell, N. (2000). Methods for distributed information retrieval. Ph. D. thesis, The Australian Nation University.

Diaz, F. and Arguello, J. (2009). Adaptation of offline vertical selection predictions in the presence of user feedback. In Proceedings of the 32nd international ACM SIGIR Conference on Research and Development in information Retrieval.

D'Souza, D., Thom, J., and Zobel, J. (2000). A comparison of techniques for selecting text collections. In Proceedings of the 11th Australasian Database Conference

French, J.C., Powell, A.L., Callan, J., Viles, C.L., Emmitt, T., Prey, K.J., and Mou, Y. (1999). Comparing the performance of database selection algorithms. In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.

Fuhr, N. (1999). A Decision-Theoretic approach to database selection in networked IR. *ACM Transactions on Information Systems*, 17(3). (pp. 229-249).

Gravano, L., Chang, C., Garcia-Molina, H., and Paepcke, A. (1997). STARTS: Stanford proposal for internet meta-searching. In Proceedings of the 20th ACM-SIGMOD International Conference on Management of Data.

Gravano, L., Ipeirotis, P. and Sahami, M. (1999). GlOSS: Text-Source discovery over the Internet. *ACM Transactions on Information Systems.*

Hawking, D. and Thistlewaite, P. (1999). Methods for information server selection. *ACM Transactions on Information Systems*, 17(1). (pp. 40-76).

Ipeirotis, P. and Gravano, L. (2002). Distributed search over the hidden web: Hierarchical database sampling and selection. In Proceedings of the 28th International Conference on Very Large Databases (VLDB).

Kirsch, S. T. (1997). Document retrieval over networks wherein ranking and relevance scores are computed at the client for multiple database documents. U.S. Patent 5,659,732.

Larson, R. (2002). A logistic regression approach to distributed information retrieval. In Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.

Le Calv, A., Savoy, J. (2000). Database merging strategy based on logistic regression. *Information Processing & Management*, 36(3).

The lemur toolkit. http://www.cs.cmu.edu/~lemur

Lu, J. and Callan, J. (2003). Content-based information retrieval in peer-to-peer networks. In Proceedings of the 12th International Conference on Information and Knowledge Management.

Lu, Y., Meng, W., Shu, L., Yu, C. T. and Liu, K. (2005). Evaluation of results merging strategies for metasearch engines. In Proceedings of the 6th International Conference on Web Information Systems Engineering.

Meng, W., Yu, C.T. and Liu, K.L. (2002). Building efficient and effective metasearch engines. *ACM Computing Survey* 34(1).

Nottelmann, H. and Fuhr, N. (2003). Evaluating different method of estimating retrieval quality for resource selection. In Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.

Nottelmann, H. and Fuhr, N. (2003). The MIND architecture for heterogeneous multimedia federated digital libraries. ACM SIGIR 2003 Workshop on Distributed Information Retrieval.

Ogilvie, P. and Callan, J. (2001) Experiments using the Lemur toolkit. In the Proceedings of the 10th Text Retrieval Conference, TREC 2001. NIST Special Publication 500-250, pp. 103-108

Powell, A.L., French, J.C., Callan, J., Connell, M., and Viles, C.L. (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.

Shokouhi, M. (2007). Central-rank-based collection selection in uncooperative distributed information retrieval. In Proceedings of the 29th European Conference on Information Retrieval, pp.160-172.

Shokouhi, M. and Zobel, J. (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

Si, L. and Callan, J. (2003a). 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.

Si, L. and Callan, J. (2003b). A Semi-Supervised learning method to merge search engine results. *ACM Transactions on Information Systems*, 21(4). (pp. 457-491).

Si, L. and Callan, J. (2004). Unified Utility Maximization Framework for Resource Selection. In Proceedings of the 9th International Conference on Information and Knowledge Management.

Voorhees, E., Gupta, N. K., and Laird, B. J. (1995). Learning collection fusion strategies. In Proc. of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.

Xu, J. and Callan, J. (1998). Effective retrieval with distributed collections. In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.

Yuwono, B. and Lee, D. (1997). Server ranking for distributed text retrieval systems on the internet. In Proceedings of the 5th International Conference on Database Systems for Advanced Applications (DASFAA).

Zhai, C.X. and Lafferty, J. (2001). A study of smoothing methods for language models applied to ad hoc information retrieval. In Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.