Emre Can Kucukoglu, 1746239
Fatih Hafizoglu, 1746059

Resource Selection Techniques for Distributed Information Retrieval

Distributed information retrieval (federated search) is based on searching multi-collections at the same time. To achieve that, subset of collections are questioned by queries and selected collections are merged into a single list. There exist three main problems in distributed information retrieval systems, these are: resource representation, resource selection and result merging [1]. In this document, mainly resource selection techniques will be discussed.

1. Large Document Model

Large document approaches can be seen as the first generation of resource selection techniques. In this technique, resources are treated as bag-of-words (“meta-documents” which are concatenation of its documents or its sampled documents). Based on this, resource selection task is reduced to a document retrieval task where running queries on “meta-documents” results the scores of resources.

CORI**:**

One of the examples using this technique is CORI [2]. CORI uses the INQUERY IR system which ranks documents to rank its “meta-documents” created from resources. Therefore, CORI does its job fast, for N documents it runs the query for each document and gets the final results from each.

CORI uses *df.icf* instead of *tf.idf* where term frequencies replaced by document frequencies and document frequencies replaced by collection frequencies. Another advantage of CORI is shows up that there is no need for another infrastructure to retrieve documents after resource selection. The same infrastructure can be used for both procedures.

In CORI, a solution also proposed for merging the returned documents from different resource. It uses resource scores to normalize document scores. Owing to this, it can produces comparable final scores for documents to rank them properly.

2. Small Document Model

It can be said that the small document approaches are the second generation of resource selection techniques. Instead of concatenating documents, it ranks the sample documents for estimating the number of relevant documents from each source. Basic algorithm used in small-document approaches is follows:

- Rank sample documents for a given query
- Take the top n results
- Assign a score to each resource based on its documents in top n results.

The small-document approaches use the algorithm that are pretty much same as above algorithm. Differences in the approaches are derived from their way to calculate scores for each resource.

ReDDE [4]: Use only count information

ReDDE resource selection method proposed after noticing that CORI algorithm does not do well in environments that contain small and large resources. Instead of using "meta-documents" they created a way that uses the information about database size and database content. Running query in samples creates a ranked list and from this list, a score for resource is calculated by only looking number of documents from the resource in top n documents. This score is the score that shows relevancy of resource and query but it is not the final score. For finalizing score, ReDDE uses the ratio of the size of resources sampled documents to the size of resource.

$$score(R|q) = n * \frac{|R|}{|S_r|}$$

$score(R|q)$: Is the score for query q on resource R

n :Is the number of documents for resource R in top n results

$|R|$: Size of resource R (number of documents taken as size)

$|S_r|$: Is the score for query q on resource R

ReDDE.top [6]: Use relevance scores

ReDDE.top is very similar to ReDDE. The difference is that ReDDE uses the count information in top n results, however ReDDE.top uses the scores of the documents in top n results.

$$score(R|q) = [score(d_x|q) + ... + score(d_y|q)] * \frac{|R|}{|S_r|}$$

$score(d_x|q)$: Is the score for query q on document d_x that is from resource R

CRCS linear [7]: Use rank information

CRCS linear is the example method for using different kind of information to calculate score: Ranks of retrieved documents

$$score(R|q) = [(n - rank(d_x|q)) + ... + (n - rank(d_y|q))] * \frac{|R|}{|S_r|}$$

$(n - rank(d_x|q))$: Is the formula CRCS uses to define a value for document d_x from resource R based on its rank in top n results

There are also other methods that use one of the above methods to calculate their own scale factor. For example GAVG uses the relevance scores but it doesn't calculate the score like ReDDE.top.

3. Classification-Based Model

Classification-based models are used to distinguish collections by analyzing queries and collections. In algorithm named *modeling relevant document distributions*, some test queries are sent to collections to find topical relevance of collections. For a given query q, MRDD algorithm finds topically relevant document distribution by training the collection with similar queries [3]. Moreover, as a second collection fusion technique, Voorhees et al. [1995] proposed *query clustering technique*. Query clustering technique measures the quality of the

query by analyzing the number of documents retrieved from a collection for that query. If two queries retrieve too many common documents, it means that they are in same topic.

Another resource selection approach called qSim proposes that past queries can be used to rank collections for new queries [5]. qSim algorithm, first calculates the similarity measurements between the current user query and the past queries. After that it estimates the utilities of available information sources by the weighted combination of search results of past queries with respect to the query similarity measurements.

Classification-based collection selection can also be done with clustering collections for a given query. Arguello et al. [2009] have proposed that in addition to large and small document models (i.e. cori, gavg, redde.top), other features can be used to assign each resource as a selected class. In this technique, three types of features used as an evidence: *collection documents*, *topic of query* and *query-click through data* [8]. First, from sampled documents, corpus features are derived and this derivation is done with query-based sampling. CORI, Seo & Craft's geometric average (GAVG) and ReDDE.top methods are used in this step. Other two steps are using query category features and click-through features. For query category features, 166 topics are selected from Open Directory Project[1]. Then, web documents are crawled according to these categories. Classifiers which trained with crawled documents, applied to documents in centralized sample index. According to this index, query now can be classified.

$$CAT_q(y_i) = \frac{1}{Z} \sum_{d \in R^{sampled}_N} P(q|d) \left(\frac{P(y_i|d)}{\sum_{y_j \in Y} P(y_j|d)}\right)$$

where y_i is value of category feature, $P(y_i \mid q)$ is category y_i's confidence value on document d. Other features used by Arguello et al. [2009] is click-through features. In this method, user clicks on a document are seen as a surrogate for collection relevance. If we consider Q_i as all queries associated with click event on a document in collection, C_i, for all i, Q_i's indexed as a document in corpus. Then, for a given query, retrieval score of each collection used as a feature.

4. Cutoff Estimation using Shard Ranking

Until this point, methods of selecting set of resources from the all resources are mentioned. But there is also another challenge exists that how many resource should be selected in order results to as close as full-dataset retrieval. For estimating the minimum number of resources, Kulkarni et al. [9] proposes a solution for this with showing fixed cutoff either overestimates or underestimates the number of resources needed to visit for many queries. Three possible resource ranking algorithms exist in the solution for the issue and these algorithms help to specify the cutoff.

These three algorithms are Lexical SHiRE (Lex-S), Rank SHiRE (Rank-S) and Connected SHiRE (Conn-S). They have the similar operations in common like all of them use of a central sample index (CSI) and they try to create a tree hierarchy from the CSI

[1] dmoz.org

documents retrieved for a query. Their differences are the way of creating this tree. After creating the hierarchy, documents are gained importance with the number of nodes needed to visited until reaching itself, starting from the top ranked document.

Lex-S algorithm steps are as follows:

1. Get the ranked documents created with CSI,
2. Put the most ranked document into to a leaf,
3. Put the lexically most similar document into a leaf that is one node away from the last leaf,
4. Goto 3rd step.

Rank-S algorithm steps are as follows:

1. Get the ranked documents created with CSI,
2. Put the most ranked document into a leaf,
3. Put the next most ranked document into a leaf that is one node away from the last leaf,
4. Goto 3rd step.

Conn-S algorithm steps are as follows:

1. Get the ranked documents created with CSI,
2. Put the most ranked document to a leaf,
3. If the next most ranked document is from the same resource with last one, put it into a leaf on the same node,
4. If the next most ranked document is from other resources, put in into a leaf that is two node away from the last one,
5. Goto 3rd step.

In SHiRE algorithms, vote of a document to its resource in hierarchy calculated as follows:

$$Vode(d) = V_d * B^{-U}$$

$Vode(d)$: Is the vote of a document to its resource

V_d :Document score assigned by the retrieval model

B : Base of exponential function

U : Step to reach

The votes of documents to their resources are calculated. After this step, their hypothesis suggests that if a resource's estimated relevance score is converges to zero, that resource close to the minimal cutoff for query. Note that 0.0001 converges to zero in their assumption.

5. Conclusion

Resource selection is one the main discussion topic of a distributed information retrieval systems. There exist some traditional resource selection approaches, some of them compare text in query to the text in a collection and some of them reveal the number of relevant documents in a collection. These baseline methods are sensitive the collection set size and their performance varies across experimental conditions. To overcome these

drawbacks of baseline approaches, another techniques can be used together with them. Classification-based approach mentioned by Arguello et al. [2009] have many advantages over traditional methods like ReDDE and CORI. These advantages are flexibility, being easy to train and effectiveness. Furthermore, experimental results show that classification method outperforms all single-evidence baselines.

Besides, Kulkarni et al. [2012] present SHiRE shard ranking algorithms. They transform CSI document ranking into a tree structure and estimate minimal shard rank cutoff using this hierarchy. These proposed algorithms are especially good at cost efficiency. They reduced the search costs approximately a quarter of one that baseline methods caused. Rank-S algorithm is also twice as accurate as the best fixed cutoff value for topical shards.

6. References

[1] Shokouhi, Milad, and Luo Si. "Federated search." *Foundations and Trends in Information Retrieval* 5.1 (2011): 1-102.

[2] J. P. Callan, Z. Lu, and W. B. Croft. Searching distributed collections with inference networks. In SIGIR 1995, pages 21–28. ACM, 1995.

[3] Towell, Geoffrey, et al. "Learning collection fusion strategies for information retrieval." *ICML*. 1995.

[4] L. Si and J. Callan. Relevant document distribution estimation method for resource selection. In SIGIR 2003, pages 298–305. ACM, 2003.

[5] Cetintas, Suleyman, Luo Si, and Hao Yuan. "Learning from past queries for resource selection." *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, 2009.

[6] J. Arguello, F. Diaz, J. Callan, and J.-F. Crespo. Sources of evidence for vertical selection. In SIGIR 2009, pages 315–322. ACM, 2009.

[7] M. Shokouhi. Central-rank-based collection selection in uncooperative distributed information retrieval. In The European Conference on Information Retrieval, Rome, Italy, 2007.

[8] Arguello, Jaime, Jamie Callan, and Fernando Diaz. "Classification-based resource selection." *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, 2009.

[9] Kulkarni, Anagha, et al. "Shard ranking and cutoff estimation for topically partitioned collections." *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM, 2012.