Literature Review

Introduction

Merging algorithms, as an important topic in metasearch area, has drawn a lot of attention recent years. Various algorithms are proposed according to different aspects. As discussed in [1], many algorithms has been proposed to solving the result merging problems, most of which are scored based, for example, Fox and Shaw [4], developed 6 methods to normalize the overall similarity across all the individual searching systems based on SMART. David Hawking [6] and his colleague implemented a metasearch engine framework that is based on current News websites. The major contribution of their work is to propose a score based approach to fuse the ranking list from different component searching systems. They used the round-robin method as a baseline and introduced several methods using the comparable scores based on the information obtained from the collections. SM-XX, as they proposed is a strategy to rank the documents from different severs according to the comparable document sore obtained by a scoring function they proposed,

$$w_{\mathrm{ij}} = \frac{NQW_{\mathrm{i}}}{\sqrt{L_q^2 + LF_i^2}} \qquad (1)$$

, where NQW$_{\mathrm{i}}$ is the number of query words appearing in the processed field of the document i, L$_{\mathrm{q}}$ is the length (number of words) of the query, and LF$_{\mathrm{i}}$ is the length of the processed field of the document i. Another strategy is to re-rank the documents in each sever according to the document score and then apply the round-robin method to the new ranked list and is named RR-XX. As it is impractical to get the document score by fetching the whole documents. They suggested that the score could be calculated based on alternative field like title, summary and can also combined with other information like date and estimated collection statistics to promote the precision. They also conducted an experiment on the content based scoring which is to fetch the whole document. This approach, if applied appropriate weight schema, can outperform the previous algorithms. However, due to the cost of downloading and indexing, it is still not a practical way to get the merged list.

A more common method was proposed by Fox and Shaw[4], who introduced 6 equation to normalize the score.

Name	Similarity
CombMAX	MAX(Individual Similarities)
CombMIN	MIN(Individual Similarities)
CombSUM	SUM(Individual Similarities)
CombANZ	Number of Nonzero Similarities
CombMNZ	SUM(Individual Similarities)* Number of Nonzero Similarities
CombMED	MED(Individual Similarities)

Table1

These 6 algorithms, as discussed by the authors, is trying to eliminate the two types of primary errors in ranking methods, namely assigning a relatively high rank to a non-relevant document and assigning a relatively low rank to a relevant document. However, it is obvious that both CombMIN and CombMAX combination method bring opposite effect on the other aspect. For CombMIN method, it can eliminate the possibility that a non-relevant document being assigned a high rank, however, the problem that CombMAX trying to fix has been amplified. The same effect exits when applying CombMIN method. And other four combination algorithms seem to perform various in different scenarios.

Besides all these scored based algorithms, many efforts have been taken to solve the problem. Aslam and Montague proposed three algorithms in [2]. They described the previous work like CEO model, which applied Bayesian model and has never implemented, and Min, Max and Sum Models, which were discussed in the last section and later in their paper, they use CombMNZ as a criteria or baseline to evaluate the performance of their new algorithms. Other three algorithms, Averaging Models, Logistic Regression Model and Linear Combination Model were also introduced briefly. They also gave some performance judgments on these algorithms or models. Then they proposed their new algorithms and conducted experiments on these algorithms. In their paper, a voting model named Borda Count, which was popular used in Election scenario, was modified to apply to our metasearch systems. In this model, servers are acting as the role of voter, and each document is just like the “candidate”. Each server or retrieval system holds its own preference of ranking on all the documents. And then, according to their description, the top ranked documents are assigned c points for candidates whose size is c, and c-1 points for the next, etc. Then we sum the score of each document and rank them according to the score. Due to the facts that different server may has different weight on different topics, they assigned weights to each server and then modified the method to a weighted borda-fuse algorithm. This model, works quite well for the system, which has many overlap in the documents. However, in reality, the repositories may vary quite differently. In that case, each document may just appears once in each rank list, which means they may distributed evenly among all the searching systems and has a lot of documents with the same scores. So the model cannot work well in the situation when there is little intersection within documents. And as a fact, their experiments showed that the new algorithm cannot outperform the baseline algorithm, but in most case, can perform better than the best-input system.

The authors then proposed another probabilistic model named Bayes-fuse, which applied Bayes rule into the retrieval system. As discussed, the relevance of a document to a query can be estimated by the equation

$$rel(d) = \sum log \frac{\Pr[r_i(d)|rel]}{\Pr[r_i(d)|irr]} \qquad (1)$$

, where the $\Pr[r_i(d)|rel]$ represent the probability that a relevant document would be ranked at level r_i by system i, while the $\Pr[r_i(d)|irr]$ is the probability that an irrelevant document would be ranked at level r_i by system i. The model sounds reasonable in theory but hard to implement in reality. In the paper, the author used the trec_eval to calculate these probabilities by human, which definitely cannot be applied to real world scenario. It may be practical to use training data to get these probabilities in real world.

But, as the collections keep changes all the time and the training will take a vast mount of time, it’s hard to guarantee the efficiency and precision.

The last algorithm they proposed two upper bounds methods, naïve bound as well as ordered pairs bound. It seems that these two methods performs quite well over all the previous algorithms. However, the statement of implementation remains quite unclear that the implementation of so call oracle is however a mystery to readers.

Other attempts have been taken by Ellen et.al [3], who used a machine learning way to retrieval relevant documents from the different collections. They used the training queries to train a relevant document distribution model or query-clustering model. But they are more focused on how to fetch the result documents instead of the merging algorithm.

As a matter of fact, it is a tough task to judge the performance of an information retrieval system. Many metasearchers have trying to persuade that their algorithms are better than others, but none of these algorithms can perform stably in every situation. And others are trying to find good ways to do these judgments on these algorithms. For lots of algorithms developed before, it is a common way to test the performance based on the Test Collection like TREC[6]. However, as discussed by David and Paul [7], the test collection approach is lack of private data and will not evolve as a real data source. Another approach is to judge the performance based on search log like clickthrough data. This approach is based on the hypothesis that high-ranked documents tend to have more clicks. However, the hypothesis does not always hold in reality. On the other hand, the search log is not easy to achieve in an un-cooperative environment. Other methods like Human experimentation in the lab as well as naturalistic observation are widely used in the practice and they both have their drawbacks. David and Paul also implemented a tool to evaluate the performance based on embedded comparison and log analysis.

References

[1] Jadidoleslamy, H. (2011). I NTRODUCTION TO M ETA S EARCH E NGINES AND R ESULT M ERGING S TRATEGIES : A S URVEY, *1*(5), 30–40.

[2] Aslam, J. a., & Montague, M. (2001). Models for metasearch. *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '01*, 276–284. doi:10.1145/383952.384007

[3] Voorhees, E. M., & Johnson-laird, B. (n.d.). The Collection Fusion Problem.

[4] Fox, E. A., & Shaw, J. A. (1994). Combination of Multiple Searches Edward A. Fox and Joseph A. Shaw Department of Compnter Science Virginia Tech, Blacksburg, VA 24061-0106.

[5] Rasolofo, Y., Hawking, D., & Savoy, J. (2003). Result merging strategies for a current news metasearcher. *Information Processing & Management*, *39*(4), 581–609. doi:10.1016/S0306-4573(02)00122-X

[6] Voorhees, E., & Harman, D. (2005). *TREC: Experiment and evaluation in information retrieval*. Retrieved from http://www.aclweb.org/anthology/J/J06/J06-4008.pdf

[7] Thomas, P., & Hawking, D. (2006). Evaluation by comparing result sets in context. *Proceedings of the 15th ACM international conference on Information and knowledge management - CIKM '06*, 94. doi:10.1145/1183614.1183632