Using Historical Data to Enhance Rank Aggregation

Miriam Fernández, David Vallet, and Pablo Castells
Universidad Autónoma de Madrid, Escuela Politécnica Superior
Ciudad Universitaria de Cantoblanco, 28049 Madrid, Spain
{miriam.fernandez,david.vallet,pablo.castells}@uam.es

ABSTRACT

Rank aggregation is a pervading operation in IR technology. We hypothesize that the performance of score-based aggregation may be affected by artificial, usually meaningless deviations consistently occurring in the input score distributions, which distort the combined result when the individual biases differ from each other. We propose a score-based rank aggregation model where the source scores are normalized to a common distribution before being combined. Early experiments on available data from several TREC collections are shown to support our proposal.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *retrieval models.*

General Terms: Algorithms, Measurement, Performance.

Keywords: Rank aggregation, score normalization, score distribution.

1. INTRODUCTION

Rank aggregation is a pervading operation in IR technology [5]. To name a few examples, rank aggregation takes place in the combination of multiple relevance criteria in most search engines; in merging the outputs of different engines for metasearch; in the combination of query-based and preference-based relevance for personalized search [1]; or even in the combination of preferences from multiple users for collaborative retrieval. Both rank-based and score-based aggregation techniques have been explored in prior research [6]. We hypothesize that that the performance of score-based aggregation may be affected by artificial, usually meaningless deviations consistently occurring in the input score distributions, which do not affect the performance of each ranking technique separately, but distort the combined result when the individual biases differ from each other, and therefore it should be possible to improve the results by undoing these deviations.

In order to combine the scores produced by different sources, the values should be first made comparable across input systems [2], which usually involves a normalization step [5]. In prior work, normalization typically consists of linear transformations [3], and other relatively straightforward, yet effective methods, such as normalizing the sum of scores of each input system to 1, or shifting the mean of values to 0 and scaling the variance to 1 [5]. But none of these strategies takes into account the detailed distribution of the scorings, and they are thus sensitive to "noise" score biases. Furthermore, they normalize each single search result in isolation, and do not even take into account if the result is good or bad in comparison to other results from the same engine, whereby the best result of a very bad run may be assigned a similar normalized score as the best result of a very good one.

We propose an effective, low-cost aggregation model where the source scores are normalized to a common score distribution, using a wider perspective of the historic behavior and particular trends of each rank source. Early experiments on available data from several TREC collections are shown to support our proposal.

SIGIR'06, August 6–11, 2006, Seattle, Washington, USA.
ACM 1-59593-369-7/06/0008.

2. SCORE NORMALIZATION MODEL

In our approach, we first generalize the notion of rank source as follows. Rather than considering a rank source as an input list of information objects, we define it by (we identify it to) a scoring function $s : \Omega_s \rightarrow \mathrm{P}$, where Ω_s is any retrieval space on which s is defined. For instance, for a search engine on a document collection Δ, we would define $\Omega_s = \Delta \times \Theta$, where Θ is the set of all queries for the engine, and $s(d,q)$ would be a similarity measure for each $d \in \Delta$ and $q \in \Theta$. For a personalized content recommender we could take $\Omega_s = \Delta \times \mathrm{Y}$, where Y is the set of all users. We may consider $\Omega_s = \Delta \times \Theta \times \mathrm{Y}$ for a personalized search engine, and so on. In real applications, the scoring functions thus modeled are used by a) calling them on specific subsets $\Psi_s \subset \Omega_s$, to be defined as the application requires, and b) using the values $s(x)$ on $x \in \Psi_s$ to induce a total order relation $\leq_s$ (a ranking) in Ψ_s. For instance, for a search engine, we would typically take $\Psi_s = \Delta \times \{q\}$, where $q \in \Theta$ is the query entered by the user, so that $(\Psi_s, \leq_s)$ is the ranked result set for q from the search engine.

Using this notation, we state the rank fusion problem as follows. Let P be the set of rank sources to be merged, and let $\Psi \subset \underset{s \in \mathrm{R}}{\times} \Psi_s$, with $\Psi_s \subset \Omega_s$, be an arbitrary combination of input values for the rank sources, so that given $\varphi \in \Psi$, $\varphi_s \in \Psi_s$ is the input to be ranked by s. For each $\varphi \in \Psi$, we want to compute an aggregated score value $s_{\mathrm{R}}(\varphi)$ based on the individual scores $s(\varphi_s)$ for each $s \in \mathrm{P}$.

The way Ψ is selected is arbitrary and we make no assumption about it in our technique. It is up to the application to build Ψ in a senseful way. In most cases this involves a fair amount of redundancy. For instance, it is very usual that Ψ_s and even φ_s are the same for all s, e.g. when the application is a metasearch system that merges the output from different engines for a single query over the same collection. However this redundancy is irrelevant for the analysis, discussion, and definition of our theoretical model. On the other hand, it allows the maximum generality for the representation of arbitrary rank aggregation problems.

Our model has the advantage that it makes it easier to model historical scoring data, which is at the core of our approach. Specifically, we consider that the score functions s are called in succes-

sive runs on different subsets $\Psi_s \subset \Omega_s$, in a way that it is possible to collect the output values returned by s for each Ψ_s, thus building a statistic series of historical data H_s, either in advance by collecting data during a certain period of regular use of the rank source, or dynamically at runtime.

Now assuming we have a history H_s for each source s, our score normalization technique works as follows. Given $\varphi \in \Psi$, we compute the aggregated score $s_R(\varphi)$ by the following steps:

a) Normalization. $s(\varphi_s)$ is normalized by a variant of the Rank-sim method [3], where instead of using a single run Ψ_s as the ranked set of retrieval objects to get scores from, a larger set from several runs is used, namely H_s, as follows:

$$\hat{s}(\varphi_s) = \frac{\left|\left\{x \in H_s \mid x \le s(\varphi_s)\right\}\right|}{\left|H_s\right|}$$

It can be seen that $\hat{s}$ is an approximation to the cumulative distribution $F_s(s(\varphi_s)) = P(x \le s(\varphi_s))$ over H_s. This distribution may be biased by accidental characteristics of the individual scoring system, which is compensated in step b of our method.

b) Standardization. Assuming that we can define a common ideal distribution $\overline{F} : [0,1] \rightarrow [0,1]$ free of any biases or noise, the potential biases of the individual score functions are compensated by computing $\overline{s}(\varphi_s) = \overline{F}^{-1}(\hat{s}(\varphi_s))$. The choice of $\overline{F}$ is critical to our method. One possible strategy would be to use exponential and Gaussian distributions, following the studies by Manmatha et al [4]. Alternatively, our current experiment consists of approximating $\overline{F}$ as the cumulative statistical distribution obtained by a) normalizing the scores in H_s to [0,1] e.g. by standard linear normalization [3], and b) joining the normalized historical data from all the sources into a joint dataset H. Then we define $\overline{F}(t) = \frac{\left|\left\{x \in H \mid x \le t\right\}\right|}{\left|H\right|}$, and $\overline{F}^{-1}$ can be computed numerically.

c) Combination. Finally, the normalized scores are merged e.g. by a linear combination or some other score-based technique.

3. EVALUATION AND RESULTS

We have tested our techniques on four test collections, namely the Web track of TREC8, TREC9, TREC9L, and TREC2001. For the comparative evaluation we have tried our technique with two reference combination functions after the normalization step, to which we will refer as: a) DCombSUM, where the fused score is computed as $s_R(\varphi) = \sum_{s \in R} \overline{s}(\varphi_s)$, i.e. our score normalization step is followed by the so-called CombSUM method [5]; and b) DCombMNZ, where $s_R(\varphi) = h(\varphi, R) \sum_{s \in R} \overline{s}(\varphi_s)$, and $h(\varphi,P) = \left|\left\{s \in R \mid s(\varphi_s) > 0\right\}\right|$, a technique named as CombMNZ in prior work [5].

We have compared these functions with other ones where the same combination step is used, but a different normalization method is applied. As a benchmark for comparison, we have taken the results published in [6], which we label as SCombSUM (CombSUM with standard score normalization), RCombSUM (CombSUM with Rank-sim normalization), and SCombMNZ (CombMNZ with standard score normalization). Table 1 shows the average results over the four collections. It can be seen that both DCombSUM and DCombMNZ are globally better that the other techniques. Although we only show the averaged results, this behavior is consistent over the four collections.

Table 1. Average precision for 10 trials of the combination of 2 to 12 ranked lists, averaged over the 4 TREC collections.

	2	4	6	8	10	12	Avg
SCombSUM	0.2598	0.2886	0.3084	0.3172	0.3204	0.3241	0.3031
RCombSUM	0.2567	0.2884	0.2847	0.2877	0.2971	0.2994	0.2857
SCombMNZ	0.2599	0.2884	0.3058	0.3176	0.3156	0.3231	0.3017
DCombSUM	0.2614	0.2942	0.3096	0.3184	0.3237	0.3268	0.3057
DCombMNZ	0.2637	0.2979	0.3090	0.3194	0.3228	0.3268	0.3066

Based on the same data, Figure 1 gives an idea of the size of historical data in H_s needed for the method to reach a good performance. It can be seen that the requirements are far from expensive.

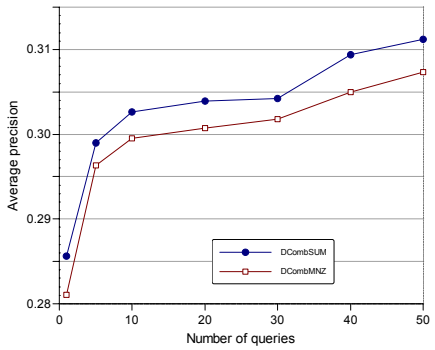

Figure 1. Number of runs needed to reach performance.

4. ACKNOWLEDGMENTS

This research was supported by the EC (FP6-027685 MESH) and the Spanish Ministry of Science and Education (TIN2005-06885).

5. REFERENCES

[1] Castells, P. et al. Self-Tuning Personalised Information Retrieval in an Ontology-Based Framework. *1st IFIP Intl. Workshop on Web Semantics (SWWS 2005)*. LNCS Vol. 3532. Agia Napa, Cyprus, 2005, 455-470.

[2] Croft, W. B. Combining approaches to information retrieval. *In: Advances in Information Retrieval: Recent Research from the Center for Intelligent Information Retrieval.* Kluwer Academic Publishers, 2000, 1-36.

[3] Lee, J. H. Analysis of multiple evidence combination. *20th ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR 97)*. New York, 1997, 267-276.

[4] Manmatha, R., Rath, R., Feng, F. Modeling score distributions for combining the outputs of search engines. *24th ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001)*. New Orleans, LA, 267-275.

[5] Montague, M., Aslam, J.A. Relevance score normalization for metasearch. *10th Conf. on Information and Knowledge Management (CIKM 2001)*. Atlanta, GA, 2001, 427-433.

[6] Renda, M. E., Straccia, U. Web metasearch: rank vs. score based rank aggregation methods. *ACM symposium on Applied Computing*. Melbourne, Florida, 2003, 841-846.