A Unified Model for Metasearch and the Efficient Evaluation of Retrieval Systems via the Hedge Algorithm *

Javed A. Aslam, Virgiliu Pavlu, Robert Savell
Department of Computer Science
Dartmouth College
{jaa,virgilpavlu,rsavell}@cs.dartmouth.edu

ABSTRACT

We present a unified framework for simultaneously solving both the pooling problem (the construction of efficient document pools for the evaluation of retrieval systems) and metasearch (the fusion of ranked lists returned by retrieval systems in order to increase performance). The implementation is based on the **Hedge** algorithm for online learning, which has the advantage of convergence to bounded error rates approaching the performance of the best linear combination of the underlying systems. The choice of a loss function closely related to the average precision measure of system performance ensures that the judged document set performs well, both in constructing a metasearch list and as a pool for the accurate evaluation of retrieval systems. Our experimental results on TREC data demonstrate excellent performance in all measures—evaluation of systems, retrieval of relevant documents, and generation of metasearch lists.

Categories and Subject Descriptors:

H.3.3 [**Information Search and Retrieval**]: Retrieval Models.

General Terms: Algorithms, Theory.

Keywords: Metasearch, Pooling, Retrieval Systems.

1. INTRODUCTION

In the annual TREC competition, participating systems return ranked lists of up to 1000 documents for 50 queries, based on estimated relevance of documents. With a document space on the order of millions, heuristics must be employed for limiting the size of pools of documents to be judged (pooling techniques). The current technique utilized by TREC (**Depth-100** pooling) places the top 100 documents from each system in the pool, and assumes any document returned below level 100 to be irrelevant. Nevertheless, pool sizes remain daunting. It has been demonstrated that stronger heuristic methods may greatly curtail the number of relevance judgments required to achieve accurate system evaluations [4, 1].

*This work partially supported by NSF Career Grant CCR-0093131 and NSF Grant 5-36955.

SIGIR'03, July 28–August 1, 2003, Toronto, Canada.

2. IMPLEMENTATION

Noting that a system performance measure suitable for evaluation of retrieval systems serves as a natural loss function for efficient online learning of system weights in a metasearch context, we present a unified method for solving metasearch and pooling based on an online algorithm for optimal allocation of resources, given the judgments of a group of expert predictors. Our method is an application of the **Hedge** algorithm, the exact details of which may be found in [3]. The goal of **Hedge** is to allocate a quantity of resources among a group of experts in proportion to the accuracy of each expert. In the online metasearch context, the similarity of the two problems is clear. Documents not yet judged (resources) are ranked for inclusion in the metasearch list according to a weighted linear combination, with weights reflecting the current performance of the retrieval systems. In the pooling context, choice of a precision measure closely related to common performance measures for evaluation of retrieval systems (such as average precision) ensures that systems are accurately differentiated given a very small and efficient collection of relevance judgments.

Hedge begins with a uniformly distributed vector of system weights. At each round, **Hedge** draws a new document to be labelled from the pool. The document's relevance judgment and per system rank is used to assess a loss to each system, and the vector of system weights is updated to reflect these losses. **Hedge** bounds are given in terms of performance on an arbitrary sequence of documents. The total loss incurred by the **Hedge** algorithm, L_H, is bounded by the loss of the best expert, L_*, and the log of the total number of experts, N: $L_H \leq c_1 L_* + c_2 \ln N$.

To adapt **Hedge** to the problem at hand, we define a loss function and a method for selecting the next document to be judged (the pooling method). The loss function is designed to reflect a document's complete contribution to a system's *total precision* (TP)—the sum of the precisions at all document levels. It is defined for document d_i at rank

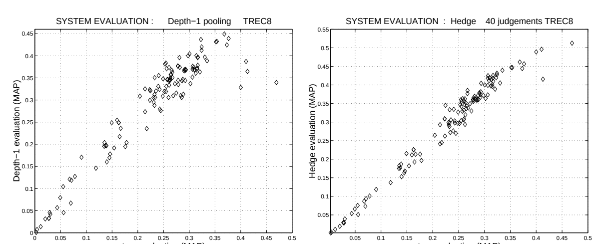

Figure 1: Depth-1 and Hedge-40 pooling assessments vs. actual TREC assessments (TREC 8).

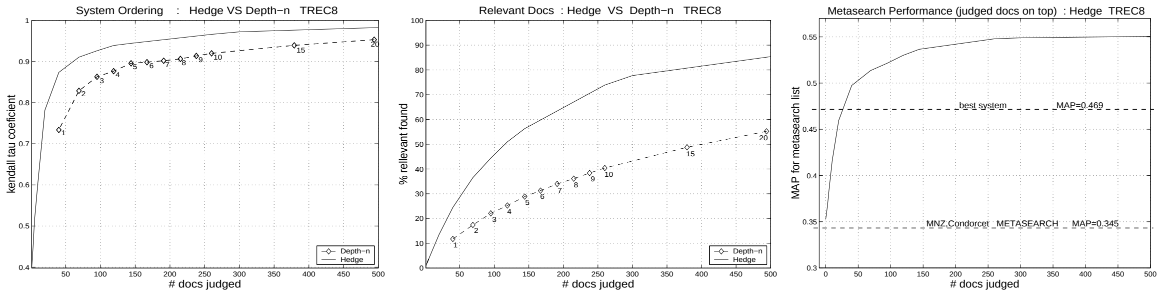

Figure 2: (a) Hedge-m and Depth-n vs. actual ranks: k-τ. (b) Hedge-m vs. Depth-n: percent of total relevant documents discovered. (c) Hedge-m: metasearch performance.

r_i by: $\ell_i = \frac{1}{2} \cdot (-1)^{I(\text{rel}(d_i))} \cdot \sum_{r=r_i}^{r_{\max}} \frac{1}{r}$, where $I(\text{rel}(d_i))$ is an indicator function for the relevance of document d_i. In the limit of complete relevance judgments, the total loss of a system converges to the negative of the total precision plus a system-independent constant. For our purposes, this measure demonstrates a close empirical relationship to other popular measures of performance such as average precision at relevant documents.

We implement a simple pooling strategy designed to maximize the learning rate of the **Hedge** algorithm. That is, at each iteration, select the document which would maximize the expected loss if it were non-relevant. Since this is exactly the unlabelled document with the maximum expectation of relevance as voted by a weighted linear combination of the systems, the strategy is also appropriate for selecting documents to be output in a metasearch list.

3. RESULTS

The **Hedge** algorithm demonstrated uniformly excellent performance across all TRECs tested (TRECs 7,8,9); in what follows, we present results from TREC 8.

To assess the quality of the pools generated by **Hedge**, we evaluate the underlying retrieval systems using standard TREC-style depth-k pools and **Hedge** pools of an equivalent size. In the following, **Depth-n** refers to an evaluation with respect to a TREC-style pool of all documents retrieved by some system at depth n or less, and **Hedge-m** refers to an evaluation with respect to the pool generated by the **Hedge** algorithm after judging m documents. Standard TREC routines were used to evaluate the systems with respect to these pools, yielding mean average precision (MAP) scores for the underlying systems as well as the rankings of those systems induced by these scores.

The scatter plots shown in Figure 1 compare the performance assessments produced by both the **Depth-1** and **Hedge-40** pools to actual TREC MAP values. The **Depth-1** plot exhibits the characteristic tail associated with the best systems which are typically scored poorly by methods relying on limited numbers of relevance judgments. As seen in the second scatter plot, this aberration is almost completely corrected by the **Hedge** algorithm after an equivalent pool size of 40 judgments. In contrast, our experiments show this tail to be prevalent in TREC-style pooling as deep as **Depth-6** (167 judgments).

Figure 2(a) compares the system rankings produced by the **Hedge** algorithm against those of **Depth-n** pooling at equivalent levels of judged documents using the Kendall's τ measure. At 40 documents, the τ for **Hedge** is 0.87 vs. 0.73 for **Depth-1**—a substantial improvement. **Hedge-69** achieves an accuracy of 0.91, vs. a **Depth-2** accuracy of 0.73. More indicative of the performance gains, however, is a comparison of the number of judgments required to achieve a particular accuracy level. For example, to achieve an accuracy of 0.87 (**Hedge-40**), the pooling method requires 95 judgments. An accuracy of 0.91 (**Hedge-69**) corresponds to a system approaching **Depth-8** (198 judgments).

Figure 2(b) compares the percentage of total relevant documents found by **Hedge** vs. **Depth-n** pooling. Again, examining the **Depth-1** system vs. **Hedge-40**, **Hedge** has found 24 percent of relevant documents vs. only 11 percent for **Depth-1**. **Hedge** maintains a very high return ratio across the plot range. Examining the curves relative to an invariant, we see that the **Depth-n** method requires approximately 104 judgments to match the **Hedge-40** return rate. The **Hedge-69** rate (36 percent) is unmatched until approximately **Depth-8** (199 judgments).

Finally, we examine the performance of the **Hedge** system as an evolving metasearch list. At each iteration, the document chosen to be judged is the one with the highest expectation of relevance. Thus, it is appropriate to build an online metasearch list from these selections. To complete the metasearch list, the remaining documents are likewise ranked by weighted linear combination. Figure 2(c) demonstrates the rapid convergence of this system to an accuracy greater than that of the best underlying system. Metasearch scores from the well known CombMNZ and Condorcet methods (equivalent in this TREC) provide a baseline MAP value of 0.345 for performance in the absence of relevance judgments. Interestingly, **Hedge** online metasearch begins with an accuracy slightly higher than this value, and then, as expected, rapidly surpasses the performance of the best system (0.469), with an MAP score of 0.497 after only 40 judgments. In the limit, the technique far surpasses the accuracy of the best system, with a final MAP of 0.55.

4. REFERENCES

[1] G. V. Cormack, C. R. Palmer, and C. L. A. Clarke. Efficient construction of large test collections. In Croft et al. [2], pages 282–289.

[2] W. B. Croft, A. Moffat, C. J. van Rijsbergen, R. Wilkinson, and J. Zobel, editors. *Proceedings of the 21th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Melbourne, Australia, Aug. 1998. ACM Press, New York.

[3] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, Aug. 1997.

[4] J. Zobel. How reliable are the results of large-scale retrieval experiments? In Croft et al. [2], pages 307–314.