

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,vip,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. Choice of a loss function closely related to the Average Precision measure of system performance ensures that retrieved documents perform well, both as a metasearch list and as a pool for accurate evaluation of retrieval systems. Application of the algorithm to TREC competition data demonstrates 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 judgements required to achieve accurate system evaluations [4, 1].

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

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGIR '03, July 28–August 1, 2003, Toronto, Canada.
Copyright 2003 ACM 1-58113-646-3/03/0007 ...\$5.00.

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 judgements of a group of expert predictors. Our method is an application of the 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 judgements.

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 judgement 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 algorithm $L_B = \ell_1, \dots, \ell_n$, is bounded by the loss of the best expert and the log of the total number of experts (N): $L_B \leq c \min_s L_s + a \ln N$.

To adapt **Hedge** to the problem, 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 r_i by:

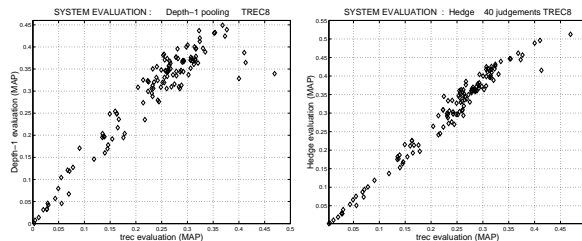


Figure 1: Depth-1 and Hedge-40 pooling vs. actual rankings—Trec 8.

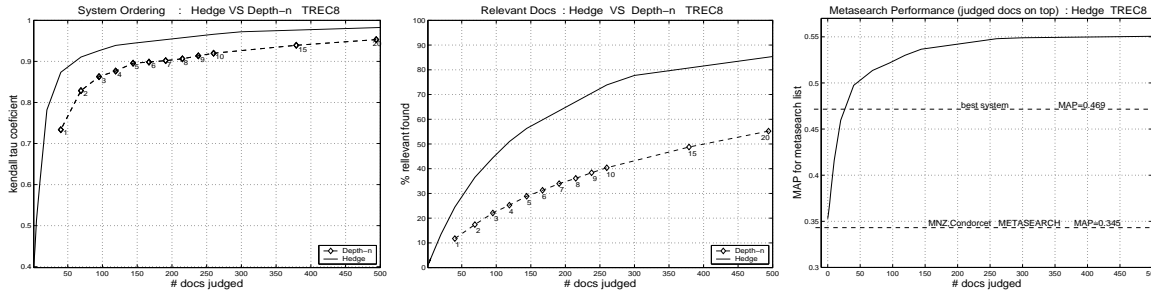


Figure 2: (a) Hedge-m and Depth-n vs. actual ranks: $k\text{-}\tau$. (b) Hedge-m vs. Depth-n: percent of total relevant documents discovered. (c) Hedge-m: metasearch performance.

$l_i = \frac{1}{2} \cdot (-1^{If(relevant(d_i))} \cdot \sum_{r:r_i \dots r_{max}} \frac{1}{r})$. In the limit of complete relevance judgements, the total loss of a system converges to the Total Precision plus a constant. The measure demonstrates close empirical relationship to other popular measures of performance, such as Average Precision—the average of the precision measurements evaluated at each relevant document in the ranked list.

We implement a simple pooling strategy designed to maximize the learning rate of the Hedge algorithm. That is, at each iteration, select the document with maximum expected loss, assuming the document is non-relevant. Since this is exactly the unlabelled document with the maximum expectation of relevance as voted by a weighted linear combination of the document’s rankings, 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 TREC8s tested (TREC 3,5,6,7,8,9). We present results from TREC 8.

To evaluate Hedge performance as a retrieval system, we compare lists of retrieved documents in depth pools of equivalent size, using the standard TREC evaluation technique. In the following, Depth-n refers to the evaluations of a TREC-style pool of all documents returned to depth n , and Hedge-m refers to the system produced by the Hedge algorithm after judging m documents. Ranking of individual retrieval systems is performed, likewise, via the standard TREC evaluation on the pools returned by Depth-n and Hedge-m techniques.

Scatter plots in Figure 1 compare the performance scores produced by both the Depth-1 evaluations and the Hedge-40 method to actual TREC rankings. The Depth-1 plot demonstrates the characteristic tail associated with the rankings of the best systems—which are typically scored poorly by methods relying on limited numbers of relevance judgements. As seen in the second scatter plot, this aberration is almost completely corrected by the Hedge algorithm after an equivalent pool size of 40 judgements. In contrast, our experiments show the tail to be prevalent in TREC-style pooling as high as Depth-6 (167 judgements).

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 judgements required to achieve a particular accuracy level. For example, to achieve an ac-

curacy of 0.87 (Hedge-40), the pooling method requires 95 judgements. An accuracy of 0.91 (Hedge-69) corresponds to a system approaching Depth-8 (198 judgements).

Figure 2(b) compares the percentage of relevant documents returned 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 judgements to match the Hedge-40 return rate. The Hedge-69 rate (36 percent) is unmatched until approximately Depth-8 (199 judgements).

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 value of 0.345 for accuracy in the absence of relevance judgements. 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 judgements. 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.