

Bayes Optimal Metasearch: A Probabilistic Model for Combining the Results of Multiple Retrieval Systems*

Department of Computer Science Technical Report TR2000-382

Javed A. Aslam Mark Montague
Department of Computer Science
Dartmouth College
6211 Sudikoff Laboratory
Hanover, NH 03755
{jaa, montague}@cs.dartmouth.edu

Abstract

We introduce a new, probabilistic model for combining the outputs of an arbitrary number of query retrieval systems. By gathering simple statistics on the average performance of a given set of query retrieval systems, we construct a Bayes optimal mechanism for combining the outputs of these systems. Our construction yields a metasearch strategy whose empirical performance nearly always *exceeds* the performance of any of the constituent systems. Our construction is also *robust* in the sense that if “good” and “bad” systems are combined, the performance of the composite is still on par with, or exceeds, that of the best constituent system. Finally, our model and theory provide theoretical and empirical avenues for the improvement of this metasearch strategy.

1 Introduction

Numerous query retrieval systems have been developed both in academia [9] and in industry (Alta Vista, Lycos, HotBot, etc.). In practice, no one system performs “better” than each of the others under all circumstances, and the “best” system for a particular task may not be known *a priori*. As such, *metasearch engines* (such as Dogpile, ProFusion, SavvySearch, etc.) have been introduced which query a number of search engines, merge the lists of pages returned, and present a resulting ranked list to the user. Our work concerns itself with precisely how to best merge the lists of ranked documents returned by different sub-engines, sometimes called the problem of “data fusion.”

Fox and Shaw [1] experiment with a group of ranked-list combination rules in the TREC competitions. Their intuitively motivated rules are based on the unweighted min, max, or sum of each document’s relevance scores over the constituent systems.

Lee [4] performs experiments with Fox and Shaw’s algorithms, arguing that “different runs retrieve similar sets of relevant documents but retrieve different sets of non-relevant documents.” He further argues that Fox’s best combination rule, known as CombMNZ, appropriately takes advantage of this feature of variant systems’ ranked lists. He also experiments with using document ranks to simulate relevance scores instead of using the relevance scores directly—we call this variant r-CombMNZ.

Vogt, et al. [7] develop a system that learns weights to linearly combine ranked lists based on a number of properties of the lists. A document’s final score (by which it is ranked) is a weighted sum of its scores in the constituent systems. Within the framework of linear combinations, they provide a formula for optimally combining two systems.

*This work generously supported by NSF grants EIA-9802068 and BCS-9978116.

The data fusion paradigm is related to a family of problem-solving techniques which consult a board of “experts” in order to make predictions as good as or better than any one of the experts alone. The learning community has used such techniques for years, including the Weighted Majority algorithm [5] and methods of boosting “weak” learners to cultivate and combine a pool of experts [3]. Freund, et al. [2] apply a variant of their AdaBoost algorithm to a family of problems closely related to our own. Intuitively, our problem is an appropriate use of such techniques, given the wide variety of retrieval methods available, with no obvious way of combining them before rankings are produced. Indeed, as Vogt, et al. [8] point out, the TREC evaluation method of “pooling” the results submitted by all participants in order to find relevant documents is itself an instance of the data fusion search methodology.

We propose a new, probabilistic model for combining the ranked lists of documents obtained by any number of query retrieval systems in response to a given query. Unlike most existing combination strategies, ours makes use of some knowledge of the average performance of the constituent systems. Our strategy most often yields a combination system whose performance *exceeds* that of any of its constituent systems. Furthermore, our strategy is *robust* in the sense that the resulting system’s performance does not appreciably degrade even when some constituent systems are quite poor. Finally, our model and theory provide avenues for the possible improvement of our proposed system.

In the sections that follow, we first propose a model for combining the ranked lists of an arbitrary number of query retrieval systems, and we derive a Bayes optimal strategy for doing so. We then detail a number of experiments with TREC data which demonstrate the aforementioned properties of our proposed strategy. Finally, we conclude with a brief discussion of these results.

2 A Probabilistic Model for Metasearch

We assume that our metasearch system has access to the ranked lists of documents produced by a given set of retrieval systems in response to a given query. (Unlike many metasearch strategies, our system does *not* require access to the actual relevance scores used to produce these ranked lists, if available.) We also assume access to some simple statistics about the average performance of the constituent systems. Given this information, we develop our probabilistic model and derive a Bayes optimal strategy for metasearch as follows.

Given the ranked lists of documents returned by a set of n retrieval systems, let $r_i(d)$ be the *rank* assigned to document d by retrieval system i (a rank of ∞ may be used if document d is not retrieved by system i). This constitutes the *evidence of relevance* provided to the metasearch strategy concerning document d . For a given document, let

$$\begin{aligned} P_{\text{rel}} &= \Pr[\text{rel}|r_1, r_2, \dots, r_n] \quad \text{and} \\ P_{\text{irr}} &= \Pr[\text{irr}|r_1, r_2, \dots, r_n] \end{aligned}$$

be the respective probabilities that the given document is *relevant* and *irrelevant* given the rank evidence r_1, r_2, \dots, r_n . The Bayes optimal decision rule for determining the relevance of a document dictates that a document should be assumed relevant if $P_{\text{rel}} > P_{\text{irr}}$ and irrelevant otherwise. Since we are interested in *ranking* the documents, we shall compute the *odds* of relevance

$$O_{\text{rel}} = P_{\text{rel}}/P_{\text{irr}}$$

and rank documents according to this measure. Applying Bayes rule, we obtain

$$\begin{aligned} P_{\text{rel}} &= \frac{\Pr[r_1, r_2, \dots, r_n|\text{rel}] \cdot \Pr[\text{rel}]}{\Pr[r_1, r_2, \dots, r_n]} \quad \text{and} \\ P_{\text{irr}} &= \frac{\Pr[r_1, r_2, \dots, r_n|\text{irr}] \cdot \Pr[\text{irr}]}{\Pr[r_1, r_2, \dots, r_n]}. \end{aligned}$$

While the $\Pr[r_1, r_2, \dots, r_n]$ term would be difficult to assess in practice, it is eliminated in our odds formulation

$$O_{\text{rel}} = \frac{\Pr[r_1, r_2, \dots, r_n|\text{rel}] \cdot \Pr[\text{rel}]}{\Pr[r_1, r_2, \dots, r_n|\text{irr}] \cdot \Pr[\text{irr}]} \tag{1}$$

By making the common *naive Bayes* independence assumptions,¹ we may rewrite this formula in a particularly simple form

$$O_{\text{rel}} = \frac{\Pr[\text{rel}] \cdot \prod_i \Pr[r_i|\text{rel}]}{\Pr[\text{irr}] \cdot \prod_i \Pr[r_i|\text{irr}]}.$$
 (2)

Finally, since we are solely concerned with *ranking* the documents, we may drop the common $\Pr[\text{rel}]/\Pr[\text{irr}]$ term and take logs, obtaining our relevance formula

$$\sum_i \log \frac{\Pr[r_i|\text{rel}]}{\Pr[r_i|\text{irr}]}.$$
 (3)

Note that $\Pr[r_i|\text{rel}]$ is the probability that a relevant document would be ranked at level r_i by system i . Similarly, $\Pr[r_i|\text{irr}]$ is the probability that an irrelevant document would be ranked at level r_i by system i . Thus, to obtain the relevance of a document for ranking purposes, we simply sum the log of the ratio of these probabilities over all systems.

3 Experiments

We evaluated our system using three data sets taken from the adhoc track of the TREC3 and TREC5 competitions. Our first data set consists of the 40 retrieval systems that were entered in the TREC3 competition, each submitting runs corresponding to 50 queries (topics 151 to 200). The 11 point average precisions for each of these systems is given in Figure 1 (together with other results), sorted according to performance.

Our second data set consists of the 61 retrieval systems that were entered in TREC5, each one over all 50 queries (topics 251 to 300). The 11 point average precisions for each of these systems is shown in Figure 2.

Our third data set consists of a subset containing 10 of the 61 the TREC5 systems, and a subset containing 10 of the 50 TREC5 queries. This subset was chosen by Vogt [6] to highlight the strengths of the metasearching technique: it contains retrieval systems chosen to maximize diversity, as measured by 9 similarity measures. The systems are: CLTHES, DCU961, anu5aut1, anu5man6, brkly17, colm1, fsclt4, gm96ma1, mds002, and uwgcx0. The queries were chosen for their large number of relevant documents: queries 257, 259, 261, 272, 274, 280, 282, 285, 288, and 289. The 11 point average precisions for each of these systems is shown in Figure 3.

We used the results of the supplied `trec_eval` program to obtain the data necessary to infer the probabilities used in Equation 3. An example of the output of this program for the best system in that year’s competition, `inq102`, is given in Figure 4. In particular, we used the average precisions at various document levels together with available information on average numbers of relevant and irrelevant documents to estimate the probabilities that relevant and irrelevant documents would be ranked at any level i , as required in Equation 3. We note that the relevance judgements used by the `trec_eval` program were made by human assessors. This technique could also be used to assess the performance of real-world systems; automatic techniques might also be employed. Details are provided in the full paper.

Note that gathering statistics about a retrieval system can be viewed as a type of “training” for our combination strategy. Therefore, in all our experiments, we first gathered all necessary statistics using the *odd* TREC topics and tested our system using the *even* topics. We then gathered all necessary statistics using the *even* TREC topics and tested our system using the *odd* topics. The performance recorded in all cases was the average of these two runs.²

In our first set of experiments, we used our proposed strategy to combine the best i retrieval systems. In other words, we used our proposed strategy to combine the top two systems, then the top three systems, and so on, culminating in the final experiment in which we combined all systems. Our results are shown in Figures 1, 2, and 3. For the purposes of comparison, we ran identical combination experiments using the well studied CombMNZ and r-CombMNZ strategies [1, 4]. Note that the performance of our strategy always

¹While these independence assumptions are almost certainly not true (for example, we are assuming that the ranks assigned to a given relevant document by two systems are independent), they are often made in practice. A more sophisticated approach to evaluating Equation 1 will almost certainly yield better performance; we discuss such possibilities in Section 4.

²This is essentially two-way hold-out cross-validation.

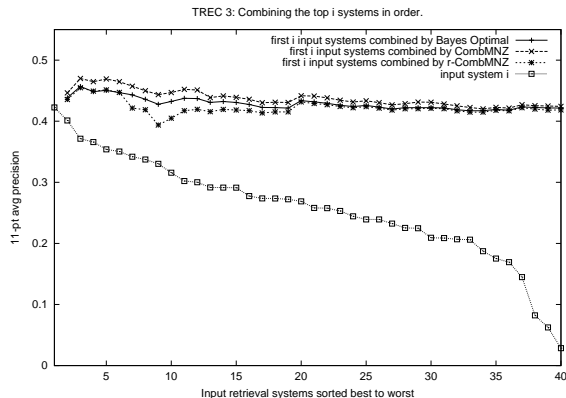


Figure 1: The 40 systems were first sorted according to performance. The x axis corresponds to system performance rank, and the y axis corresponds to 11 point average precision. The lowest curve shows the performance of each of the 40 systems entered in TREC3. In another plot, we show the performance of the CombMNZ strategy in combining the top i ranked systems. In a third plot, we show the performance of our metasearch strategy in combining the top i ranked systems. Finally, the performance of the r-CombMNZ strategy is shown.

equalled or exceeded the performance of the *best* constituent system, even when extremely poor systems were part of the combination. This was also true of the CombMNZ strategy, but not of r-CombMNZ.

On the TREC3 data set the performance of our strategy peaked when combining the top three systems, at which point it yielded an 8.1% improvement over the best system and a 14.7% improvement over the average performance of the three systems it was combining. Note that the performance of the best system was quite good; in other experiments, we have seen far greater absolute and percentage gains when combining systems of more moderate performance. Also note that while our strategy takes advantage of training data (which CombMNZ does not), CombMNZ takes advantage of relevance scores (which Bayes Optimal does not). Without the relevance scores, our model is consistently as good or better than the MNZ strategy. While training data can be generated by the metasearch implementor, if relevance scores are not provided by the constituent retrieval systems, there is often no way to gain access to them.

On the third data set the performance of all three systems continues to improve as more systems are added, and our Bayes Optimal procedure eventually edges out the MNZ strategies. This might be expected on a data set that was chosen for its diversity—on such a data set, the independence assumptions made by the Bayes Optimal model are better satisfied than on the other data sets.

In our second set of experiments we fused the *worst* i retrieval systems. See Figures 5, 6, and 7. It is surprising how much improvement metasearch shows over the constituent systems when all of the systems being fused are poor. On the TREC3 data set the advantage of using training data is visible in the superior performance of the Bayes Optimal strategy when only a few systems are being combined, some of which are significantly worse than the others.

In our final set of experiments, we examined the performance of the three metasearch strategies when combining random groups of retrieval systems. Each data point in Figures 8, 9, and 10 represents the average value obtained over at least 100 experiments performed as follows: Select n (for $n \in 1...8$) input systems randomly, fuse them, and report the resulting list's 11 point average precision. We found that each fusion algorithm was usually able to meet or exceed the performance of the best system being fused, and far outperformed the average system being fused. We also found that performance continues to go up as more systems are being combined, though the performance improvements slow after 5 or so systems. Finally, the Bayes Optimal strategy outperformed the MNZ strategies when the inputs are diverse.

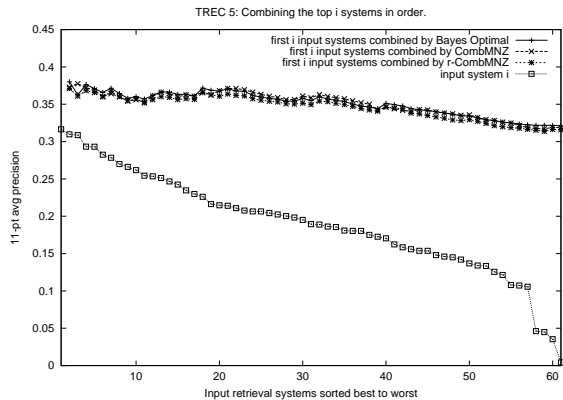


Figure 2: The same experiment as in Figure 1, but over the 61 TREC5 retrieval systems.

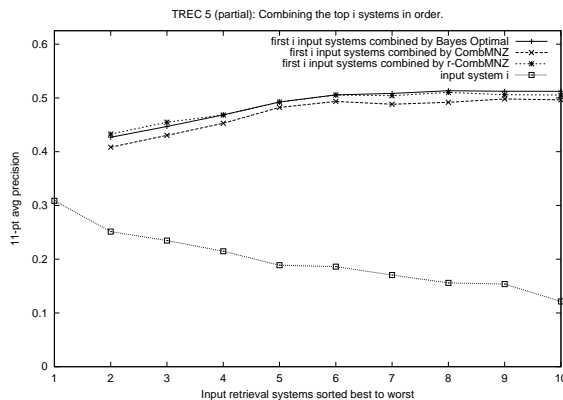


Figure 3: The same experiment as in Figure 1 and Figure 2, but over the third data set.

4 Discussion

We have proposed a probabilistic model for combining the outputs of an arbitrary number of query retrieval systems. Within this model, we have derived a Bayes optimal strategy for performing such combinations. Finally, using TREC data, we have demonstrated that our strategy is both powerful and robust.

One way in which our strategy may very likely be improved is in the (full or partial) elimination of the independence assumptions used to transition from Equation 1 to Equation 2. These independence assumptions, while commonly made in practice, are almost certainly untrue. In fact, Lee [4] argues convincingly that the sets of relevant documents returned by retrieval systems are highly correlated, while the sets of irrelevant documents returned are far less so. A more sophisticated evaluation of Equation 1 which accounts for this dependence will almost certainly yield improvements in our strategy, and we are currently pursuing just such an improvement.

References

- [1] E. A. Fox and J. A. Shaw. Combination of multiple searches. In D. Harman, editor, *The Second Text REtrieval Conference (TREC-2)*, pages 243–249, Gaithersburg, MD, USA, Mar. 1994. U.S. Government Printing Office, Washington D.C.

```

Queryid (Num):          50
Total number of documents over all queries
Retrieved:             50000
Relevant:              9805
Rel_ret:               7305
Interpolated Recall - Precision Averages:
at 0.00                0.8992
at 0.10                0.7514
at 0.20                0.6584
at 0.30                0.5724
at 0.40                0.4982
at 0.50                0.4272
at 0.60                0.3521
at 0.70                0.2915
at 0.80                0.2173
at 0.90                0.1336
at 1.00                0.0115
Average precision (non-interpolated)
for all rel docs(averaged over queries)
                        0.4226
Precision:
At 5 docs:             0.7440
At 10 docs:            0.7220
At 15 docs:            0.6867
At 20 docs:            0.6740
At 30 docs:            0.6267
At 100 docs:           0.4902
At 200 docs:           0.3848
At 500 docs:           0.2401
At 1000 docs:          0.1461
R-Precision (precision after R
(= num_rel for a query) docs retrieved):
Exact:                 0.4524

```

Figure 4: trec_eval statistics for inq102.

- [2] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. In J. Shavlik, editor, *Machine Learning: Proceedings of the 15th International Conference (ICML '98)*, pages 170–178, Madison, Wisconsin, USA, July 1998. Morgan Kaufmann, San Francisco, California.
- [3] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Machine Learning: Proceedings of the Thirteenth International Conference*, pages 148–156, 1996.
- [4] J. H. Lee. Analyses of multiple evidence combination. In N. J. Belkin, A. D. Narasimhalu, and P. Willett, editors, *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 267–275, Philadelphia, Pennsylvania, USA, July 1997. ACM Press, New York.
- [5] N. Littlestone and M. Warmuth. The weighted majority algorithm. Technical Report UCSC-CRL-89-16, U. C. Santa Cruz, 1989.
- [6] C. C. Vogt. How much more is better? characterizing the effects of adding more ir systems to a combination. In *RIAO'2000 Conference Proceedings, Volume 1*, pages 457–475, Paris, France, Apr. 2000.
- [7] C. C. Vogt and G. W. Cottrell. Fusion via a linear combination of scores. *Information Retrieval*, 1(3):151–173, 1999.
- [8] C. C. Vogt, G. W. Cottrell, R. K. Belew, and B. T. Bartell. Using relevance to train a linear mixture of experts. In E. Voorhees and D. Harman, editors, *The Fifth Text REtrieval Conference (TREC-5)*, pages 503–515, Gaithersburg, MD, USA, 1997. U.S. Government Printing Office, Washington D.C.
- [9] E. Voorhees and D. Harman. Overview of the Eighth Text REtrieval Conference (TREC-8). In D. Harman, editor, *The Eighth Text REtrieval Conference (TREC-8)*, Gaithersburg, MD, USA, 2000. U.S. Government Printing Office, Washington D.C.

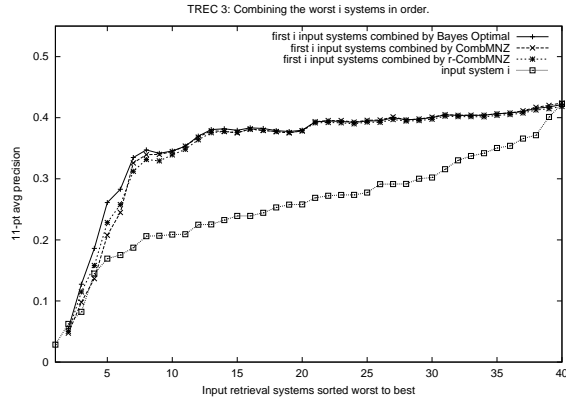


Figure 5: The 40 systems were first sorted in order of increasing performance. The x axis corresponds to system performance rank, and the y axis corresponds to 11 point average precision. The lowest curve shows the performance of each of the 40 systems entered in TREC3. In another plot, we show the performance of the CombMNZ strategy in combining the worst i ranked systems. In a third plot, we show the performance of our metasearch strategy in combining the worst i ranked systems. Finally, the performance of the r-CombMNZ strategy is shown.

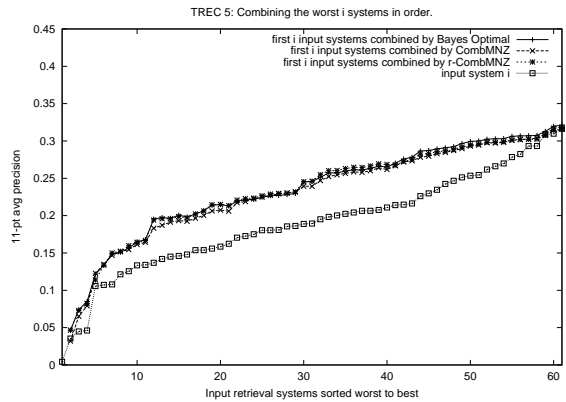


Figure 6: The same experiment as in Figure 5, but over the 61 TREC5 retrieval systems.

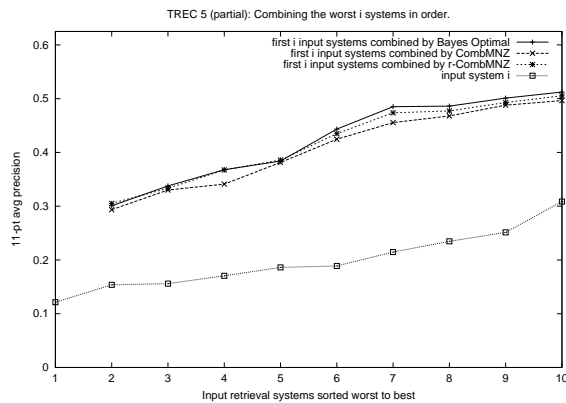


Figure 7: The same experiment as in Figure 5 and Figure 6, but over the third data set.

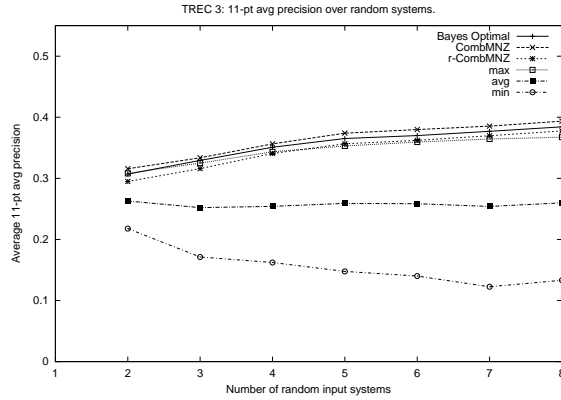


Figure 8: The x axis corresponds to the number of randomly chosen input systems that are being fused. The y axis corresponds to 11 point average precision (averaged over at least 100 trials). The curve labeled “max” shows the 11 point average precision for the system with best performance of the systems being fused, averaged over the trials. Similarly, “avg” and “min” are shown.

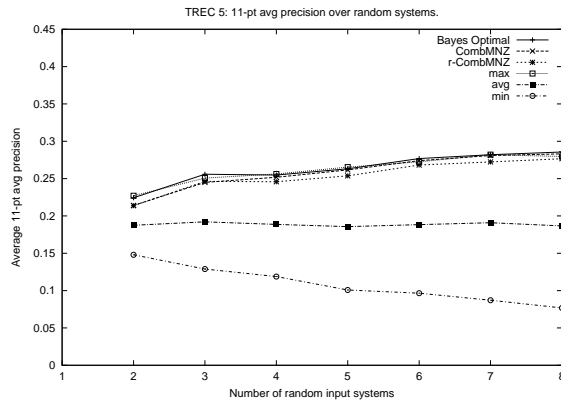


Figure 9: The same experiment as in Figure 8, but over the 61 TREC5 retrieval systems.

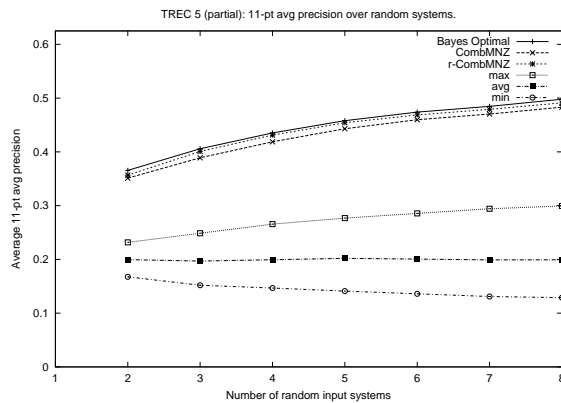


Figure 10: The same experiment as in Figure 8 and Figure 9, but over the third data set.