Automatic Combination of Multiple Ranked Retrieval Systems

Brian T. Bartell†
bbartell@eb.com

Garrison W. Cottrell‡
gary@cs.ucsd.edu

Richard K. Belew‡
rik@cs.ucsd.edu

†Advanced Technology Group
Encylopædia Britannica, Inc.
3252 Holiday Court, Ste. 208
La Jolla, California 92037

‡Department of Computer Science & Engineering-0114
University of California, San Diego
La Jolla, California 92093-0114

Abstract

Retrieval performance can often be improved significantly by using a number of different retrieval algorithms and combining the results, in contrast to using just a single retrieval algorithm. This is because different retrieval algorithms, or retrieval *experts*, often emphasize different document and query features when determining relevance and therefore retrieve different sets of documents. However, it is unclear how the different experts are to be combined, in general, to yield a superior overall estimate. We propose a method by which the relevance estimates made by different experts can be automatically combined to result in superior retrieval performance. We apply the method to two expert combination tasks. The applications demonstrate that the method can identify high performance combinations of experts and also is a novel means for determining the combined effectiveness of experts.

1 Introduction

In text retrieval, two heads are definitely better than one. Retrieval performance can be greatly improved by using a number of different retrieval algorithms (or *experts*) and combining their results, in contrast to using just a single retrieval algorithm. Each expert contributes its estimates of which documents are likely to be relevant to the user's query, and the combined set is typically more valuable than any single expert's estimates.

Part of the benefit may be due to the increased recall resulting from a combination of experts. It has been observed that different experts often retrieve different documents; e.g., Katzer et al. [10] found that very different sets of documents were retrieved using different document representation methods, and Harman [9] observed that the systems in the 1993 Text REtrieval Conference (TREC-1) retrieved substantially different sets of documents, even though most systems performed at the same approximate level. Since different experts can retrieve different documents, the perceived performance of the combined system may be higher because of the increased levels of recall that are attainable.

Not all of the performance benefit can be attributed to enhanced recall in the combined system. For example, Saracevic & Kantor [13] demonstrated that the odds of a document being relevant to a query increase monotonically with the number of experts that retrieve the document. A possible explanation is offered by a recent study by Belkin et al. [3]. This study demonstrated that the retrieval results based on different query formulations can be combined to result in performance that is better than any single formulation. Belkin et al. suggest that each expert's relevance estimates may be interpreted as sources of evidence of the true relevance. Additional experts provide additional evidence resulting in a more accurate estimate of the true probability of relevance.

The principal difficulty with combining experts, however, is in deciding how the multiple experts are to be combined to generate an overall retrieval score. In this work, we provide a method by which multiple experts can be automatically combined into a single retrieval system. The technique is applicable to *ranked retrieval* systems, in which the overall performance goal is to correctly rank the documents in the order of their relevance to the user's query. For these systems, the method automatically determines a combination of experts that is optimized to rank relevant documents before less relevant ones. We will demonstrate that this automatically derived combination of experts performs much better than any of the individual experts when applied to two different problems.

B. W. Croft et al. (eds.), *SIGIR '94*

In addition, we demonstrate that the method is a useful and novel analytic tool. Belkin et al. [3] made the important observation that a different query formulation (or expert) can contribute to improved performance in the combined system even when it performs poorly on its own. The consequences of this observation are significant: A candidate retrieval method cannot be discounted simply because it does not perform well relative to alternative methods. This same method might still enhance performance when combined with others. This poses a dilemma for system designers: How can a candidate method be evaluated to determine whether it should be added to the system, when simple evaluation of the method (with comparison to other techniques) does not predict its actual utility? The approach we present here can potentially offer insights into this dilemma, as it offers a method for evaluating the performance of experts in combination with others rather than in isolation.

In the next section, we outline our method for automatically combining experts. The approach is based on the general rank optimization method developed in previous work [1] [2]. Sections 3 and 4 present the results of two applications, Section 5 compares the method to an alternative, and Section 6 discusses the method's strengths and limitations.

2 Expert Combination Algorithm

A number of methods for combining retrieval experts have been proposed. Perhaps the most common approach is by manual search of a small set of possible combination strategies. The system designer evaluates one or more candidate combination strategies using a set of queries with relevance-tagged documents, and the strategy resulting in the best performance is selected. Typical of this approach are: Belkin et al.'s [3] analysis of alternative query formulations on the TREC database, Lewis & Croft's [11] comparison of term, phrase, and concept clustering approaches, and Turtle & Croft's [16] inference network framework, in which manual search is used to find the best weighted average of experts. A limitation of manual search is that typically only a small number of strategies are examined.

Alternatives to search include Fox et al.'s [5] Similarity Merge approach, in which a document's overall relevance score is the maximum of all expert's relevance estimates, and Thompson's [15] evaluation method in which each expert is weighted based on its individual utility. In this second approach, each expert is evaluated independently to determine its relative performance level. During retrieval, the experts' relevance estimates are combined in proportion to their performance level. Weighting experts based on their individual performance does not clearly address how they will perform together. Indeed, Thompson found that the combined experts, weighted by performance level, performed no better than a combination using uniform weights. Another alternative is Thompson's Combination of Expert Opinion (CEO) model [14]. CEO is a Bayesian model of the combination of evidence from a set of experts. Experts are combined to yield a more reliable overall estimate of a document's relevance to a query. The method assumes that the experts are statistically independent, though work continues to make this model more generally applicable [15].

Our approach differs from these approaches in that 1) a large space of possible combinations is heuristically searched, 2) the ability of the *combined* system to rank documents based on their relevance is explicitly optimized, and 3) the only restriction on the experts is that they generate numerical estimates of the relevance of documents to the query – the experts need not be statistically independent.

2.1 Parameterized Mixture of Experts

To automatically combine the retrieval experts, we define a model specifying how the experts may be combined. The model takes the individual experts' estimates of document relevance and combines them into a single estimate. In common with a number of the approaches identified above, we use a *linear* combination model throughout this paper. In the linear model, the overall estimate is the sum of scaled estimates from the individual experts. For example, the overall estimate for document d and query q for a system with 3 experts is:

$$R_{\Theta,q}(d) = \Theta_1 E_1(q,d) + \Theta_2 E_2(q,d) + \Theta_3 E_3(q,d) \tag{1}$$

where $R_{\Theta,q}(d)$ is the overall estimate, $E_i(q,d)$ is the relevance estimate of expert i, and Θ_i is the scale on expert i. The set of Θ_i, denoted by Θ, are free parameters in the model. The goal of our method is to automatically determine values for these parameters so that the overall estimates result in the best ranking of documents possible. Though we emphasize the linear model here, non-linear models (neural networks) have yielded positive results [1].

2.2 Optimizing Ranking Performance

To find values for the parameters, we use a criterion which measures how well the model is ranking documents for a set of training queries. We numerically optimize the criterion in order to heuristically search for parameter values which result in superior ranking performance. The criterion we use is a variation on Guttman's Point Alienation [8], a statistical measure of rank correlation. Our criterion is:

$$J(R_\Theta) = \frac{-1}{\mid Q \mid} \sum_{q \in Q} \frac{\sum_{d \succ_q d'} (R_{\Theta,q}(d) - R_{\Theta,q}(d'))}{\sum_{d \succ_q d'} \mid R_{\Theta,q}(d) - R_{\Theta,q}(d') \mid} \tag{2}$$

Here, Q is the set of training queries and d and d' are documents retrieved by at least one of the experts for query q. It is assumed that we either know the relevance of all retrieved documents for the training queries, or we restrict the optimization to those documents that are known. This knowledge is represented by the binary relation $\succ_q$ (a notation adopted from Wong & Yao [17]). $\succ_q$ is a preference relation over document pairs, interpreted as:

$$d \succ_q d' \iff \text{the user prefers } d \text{ to } d' \tag{3}$$

This preference relation is an alternative to the more standard two-valued relevance in which a document is either *relevant* or *irrelevant* to a query. Two-valued relevance can serve as the basis for a preference relation when such a relation can not be obtained from users. In this case, the two-valued relevance judgements are interpreted as a preference order between the relevant (preferred) set and the irrelevant (not preferred) set. The preference relations used in the experiments reported in the next sections were derived from two-valued relevance judgements in this fashion.

We use Conjugate Gradient, a gradient-based numerical optimization technique, to minimize J. Conjugate Gradient is preferred over other gradient methods because it is parameter-free for most applications and source code is readily available (see [12]). Criterion J is not differentiable everywhere (e.g., J is singular if $R_{\Theta,q}(d) = R_{\Theta,q}(d')$, for $d \succ_q d'$) and therefore might be resistant to gradient-based optimization. However, other work has demonstrated that the singularities do not generally cause difficulties [1], and Conjugate Gradient has been quite effective throughout our experiments.

The goal of the optimization is to find parameter values such that the system ranks document d higher than document d' whenever d is preferred by the user to d'. The criterion J defined in equation (2) is a particular mathematical formalization of this goal. There are numerous alternatives, however, to the particular criterion we have used. For example, other criteria have been proposed to solve similar rank-order problems in the field of Multidimensional Scaling (MDS) [4]. We have selected this variation of Guttman's Point Alienation because it is amenable to gradient-based optimization, it is general enough to handle arbitrary user preference relations, and because the measure has been shown to correspond well empirically to average precision, a more standard measure of retrieval effectiveness in Information Retrieval [2]. We have chosen not to explicitly optimize average precision because it is a discrete measure and is therefore not amenable to gradient-based optimization. However, methods do exist for optimizing such non-differentiable criteria, and some methods have been applied to tasks in Information Retrieval [7]. This may be a fruitful direction for further research.

3 Learning Weights on Phrases and Terms

Our first application of the method involves combining two experts. This application demonstrates that the method can be used to evaluate expert performance in combination with other experts, in contrast to the more standard individual analysis.

The first of the two experts (the "term" expert) is a standard vector-space retrieval system using white-space delimited terms. The second expert (the "phrase" expert) is a phrase identifier. It retrieves documents which contain phrases appearing in the query. The phrase expert identifies phrases using a trivially simple algorithm: Any bigram (adjacent word pair) not containing a high-frequency noise word and not crossing a sentence boundary is accepted as a phrase. The phrase expert will never retrieve any documents not already retrieved by the term expert (since any phrase which causes a document to be retrieved by the phrase expert is composed of terms which would cause the same document to be retrieved by the term expert). Thus, the usefulness of the phrase expert in the combined system stems solely from its ability to improve the term expert's document ranking.

Retrieval System	Average Precision Performance Avg Precision	% over Phrases
Phrase Expert	.1672	-
Term Expert	.2340	+40%

Table 1. The phrase expert has very low performance relative to the term expert.

Retrieval System	Truncated Precision for the k Top-Ranked Documents $k = 5$	$k = 10$	$k = 15$	$k = 30$
Phrase Expert	.1571	.1365	.1237	.1094
Term Expert	.2132	.1871	.1761	.1602

Table 2. The phrase expert performs worse than the term expert even for the small set of highest ranked documents retrieved.

3.1 Test Collection

To train and test this combination model, we use portions of the Encylopædia Britannica, an edited collection widely regarded as the premier English language general reference work. Articles in the Encylopædia Britannica (EB) have been semantically organized by editors into a hierarchical topic tree, called the Propædia. Our test collection is composed of the 8,006 Micropædia articles categorized by the editors as pertaining to Part 5, Human Society.

The queries used to train and test the system are also derived from the Propædia topic tree. Each node in the topic tree represents a refinement of the topic in the parent node. For example, the root node of Part 5 of the Propædia is "Human Society". Its children are "Social Groups: Peoples and Cultures", "Politics and Government", etc. Further down the tree one finds the more refined topic "Social Differentiation and Stratification". Associated with each of these nodes, both internal and leaf nodes, is a set of articles that discuss aspects of the topic of the node. This structure naturally permits interpretation of the node topics as queries, and their associated documents as the relevant set. We are able to extract 356 queries in this manner. These 356 queries have an average of 7.2 relevant documents per query. The set of 356 queries are partitioned into two subsets, the training set and the test set. The training set, with 128 queries, is used in equation (2) to optimize the model; the remaining 228 queries are used to test the performance of the optimized system.

3.2 Results

Tables 1 and 2 provide a summary of the performance (measured by average precision) of the two experts evaluated independently using the 228 test queries. As the tables illustrate, the term expert performs much better than the phrase expert, topping the phrase expert's average precision performance by 40%. We might expect that much of this effect is because the term expert retrieves many more documents, and therefore can achieve higher performance through much higher levels of recall. For example, the term expert retrieves an average of 2,019 documents per query, whereas the phrase expert retrieves only 28! However, we have found that this difference in recall is not a complete explanation. Even when only the highest ranked documents are used to examine performance, the term expert still outperforms the phrase expert. This performance is summarized in Table 2. In addition, for all of 11 levels of recall between 0.00 and 1.00 the term expert has higher precision than the phrase expert.

In light of these results, it would seem that we should readily reject this phrase expert as a candidate for inclusion in any retrieval system. After all, it does not perform very well, even on the small set of documents it does retrieve. However, the results of our optimization suggest otherwise. The optimized combined system performs 12% better than the best individual expert (the term expert). This result

Retrieval System	Optimized Performance on 228 Test Queries Avg Precision	% over Phrases	% over Terms
Phrase Expert	.1672	-	-
Term Expert	.2340	+40%	-
Optimized Combination	.2618	+57%	+12%

Table 3. The phrase expert still contributes to improved overall performance despite its very low individual performance.

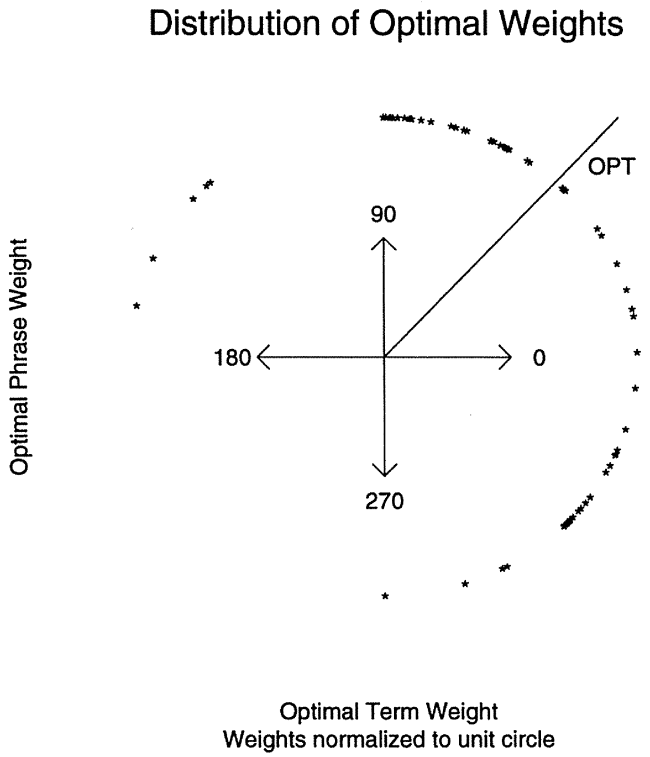

Figure 1. The optimal weights for each independent query vary greatly. Each point above identifies the optimal weights for one of the 128 queries. The optimal weights (OPT) when all 128 queries are optimized together is also shown.

is summarized in Table 3[1]. Thus, the combined system is able to use the two experts' scores to result in a ranking of documents that is better than any single expert's. Note that this comparison is made using the 228 test queries, not the 128 queries used by the method to optimize the model. Thus, the performance scores represent an estimate of how well the optimized model will perform when deployed in the real world accepting queries which it has never before seen. Interestingly, the optimized combination actually weights the phrase expert slightly higher than the term expert: The optimized phrase weight is found to be 0.738, and the term weight is 0.675. This is despite the phrase expert's much lower individual performance.

The reason the phrase expert can contribute to the overall system is evident in a query-by-query analysis. Figure 1 depicts the optimized weights for each of the 128 training queries optimized separately. The weights have been normalized to fall on the unit circle as this does not affect the rank order of documents and clarifies the analysis. The large number of points in the lower right quadrant illustrates that the optimized phrase weight is negative for almost as many queries as it is positive. This suggests that for many queries, the best strategy is to *subtract* the phrase estimate from the term estimate – that is, the phrase expert is identifying irrelevant documents better than relevant documents. However, for those queries for which the phrase parameter is positive, the ratio of the phrase parameter to term parameter is very large. Thus, for these queries, the phrase expert is *very* accurate – the best strategy is to rank documents first using the phrase expert score, and then use the term expert to rank the documents which were not retrieved by the phrase expert. In summary, the phrase expert often performs poorly; however, when the phrase expert is good, it can be very good, resulting in a net positive contribution to the model.

[1] We have used average precision scores to make comparisons between retrieval systems rather than criterion scores even though the optimization procedure only uses the criterion scores. We report average precision because it is a standard measure for comparison in Information Retrieval and because it can be difficult to draw conclusions about absolute system performance from the criterion scores (cf. [4, p. 47]).

Average Precision on Test Set, By Partition					
Query Partition	Phrase Expert	Count Expert	Vector Expert	Optimized System	Improvement over Count Expert
1	.2315	.3235	.2162	.5121	+58%
2	.2989	.3766	.3093	.5559	+48%
3	.3761	.4610	.3173	.6373	+38%
4	.2675	.3344	.2611	.5161	+54%
5	.3600	.4810	.3013	.6011	+25%
6	.1729	.3369	.3338	.5153	+53%
7	.3402	.3248	.2018	.5732	+76%
8	.3124	.4468	.2999	.5629	+26%
Average	.2949	.3856	.2801	.5592	+47%

Table 4. The optimized combination model performs much better than the best individual (the count expert).

4 Optimal Performance in a Commercial System

The second application of our method involves optimization of three experts in a commercial system, *SmarTrieve*, developed by Compton's New Media, Inc. In this application, we emphasize the performance enhancements which are attainable using the method. Part of the retrieval algorithm employed by SmarTrieve entails consulting three experts. These experts provide the following relevance estimates: A traditional vector space estimate (the vector expert); a count of the number of query terms in the document (the count expert); and an estimate of the number of query phrases in the document (the phrase expert).

4.1 Test Collection

We use the Compton's MultiMedia Encyclopedia (CMME) to train and test the combination model. The CMME consists of 20,701 documents along with additional associated multimedia objects (such as short video, sound clips, etc.). All 20,701 full-text documents are used in the experiments.

The training and test queries are derived from a set of questions found in the printed version of the CMME. In each of the 25 volumes, the editors have provided a set of questions which can be answered by one of the articles in that volume. For example, a question in volume 1 is: "What sport may have given rise to the myth of the Minotaur?" (Answer: acrobatics, found in the article titled "Aegean Civilization"). We divide these queries into 8 partitions, each partition having a set of approximately 60 training queries and a set of approximately 60 test queries. We optimize the model separately for each of the 8 partitions, and then evaluate the optimized model using the corresponding test set. The optimization is repeated over these 8 partitions in order to avoid any bias due to the particular queries in one group. To avoid local minima in training, 5 different randomly initialized models are optimized for each partition. The model with the best average precision performance on the training queries is selected as the representative for each partition. Even though we select the model which performs best on the training set, we will evaluate the model using the reserved set of test queries so as not to bias the comparison in favor of the optimized model.

When optimizing the model for each partition, we do not use the relevance judgements for all 20,701 documents. Rather, for each query we identify the 15 documents ranked highest by the unoptimized SmarTrieve system. Relevance judgements for only these top-ranked 15 documents are used for each query in training. Using only the top-ranked 15 documents simulates the scenario in which the relevance judgements are acquired from actual sessions with users of the system – it would be unlikely to convince users to provide feedback on all 20,701 documents in the collection! 15 was chosen arbitrarily as a reasonably small though nevertheless informative subset size.

4.2 Results

The performance of the optimized system and a comparison to the individual experts is illustrated in Table 4. As in the previous section, we compare the performance of the different systems using the sets of test queries and not the queries used to optimize the model. The improvement is at least 25% over the best individual expert (the count expert) for all partitions, and the average improvement is 47%. The difference in performance between the count expert and the optimized system is statistically significant (with $p < 0.0005$, determined by an analysis of variance (ANOVA) of the average precision scores across the 8 partitions). The optimized weights themselves do vary somewhat over the 8 partitions, but are generally stable. The average optimized weights (over the 8 partitions), normalized to have unit sum

squared length, are: $\Theta_{\text{Phrase}} = 0.305$, $\Theta_{\text{Count}} = 0.946$, and $\Theta_{\text{Vector}} = 0.112$. The average angle between weight vectors[2] in the different partitions is 13.9°, with standard deviation 7.9. This variation between partitions suggests that the roughly 60 training queries used in each partition do not completely determine the theoretically optimal model, in which the model performs the best on average over all possible queries. Rather, the optimized model is still somewhat specialized to the characteristics of the training set. However, the small average angle between weight vectors and the high retrieval performance on the sets of test queries indicate that different selections of training queries do not lead to wildly different or over-specialized optimized solutions.

Interestingly, the performance improvement of the optimized system is much smaller (only +6%) when the model is optimized using all retrieved documents for each query instead of just the top-ranked 15. This was initially surprising to us, since the additional documents should provide additional information with which to select good parameter values. We have found, however, that using all documents instead forces the model to overcompensate for a few relevant documents which are very poorly ranked by the unoptimized system. That is, for a few queries, the relevant documents are ranked at the end of the retrieved list instead of near the beginning. It is impossible for the linear model to compensate for these difficult queries without sacrificing performance for the other queries, so the optimized model performs unsatisfactorily overall. Restricting optimization to only the top-ranked 15 documents effectively ignores these problematic documents. This behavior appears to be a characteristic of the linear model – other work has demonstrated that some non-linear neural network models do not suffer from this sensitivity to problematic documents [1].

5 Comparison to a Supervised Learning Algorithm

There are alternatives to the particular rank order criterion we have used in this work. One supervised criterion that is very popular in pattern classification and neural networks is the squared error criterion:

$$S(R_\Theta) = \sum_q \sum_d (y_{q,d} - R_{\Theta,q}(d))^2 \tag{4}$$

where $y_{q,d}$ is 1 if document d is relevant to query q, and 0 otherwise. Minimizing this criterion will make the system's relevance estimates more like the true probabilities of relevance. A similar squared error criterion has been used recently by Fuhr & Buckley [6], though they use the criterion to optimize relevance estimates based on single terms rather than to optimize the overall retrieval performance as we do here.

We have applied the squared error criterion to the three expert task in the previous section and have found that it does not perform as well as our rank order criterion. The best improvement to average precision over the count expert, for a number of different linear and non-linear models and training environments, is 2%. A possible explanation for squared error's low performance is that it imposes constraints on the system's relevance estimates which are stronger than are needed for good ranked retrieval. The main goal in a ranked retrieval system is to order the documents in the order of their relevance. Squared error goes well beyond this goal by requiring that the system's estimate be 1.0 for all relevant documents, 0.0 for the others. Certainly any solution that satisfies these constraints also satisfies the rank constraints; however, it appears difficult to satisfy these stronger squared error constraints in practice.

6 Discussion

We have demonstrated a method for combining experts which is effective and applicable to a range of common experts in information retrieval. The method uses well understood numerical optimization techniques to optimize the combination of experts leading to an overall system specifically configured to rank relevant documents before less relevant ones. The method is an alternative to a number of techniques for combining experts that have been proposed recently, in particular the manual search method, weighting by individual performance, and Thompson's CEO method [14]. Each of these alternatives has certain strengths, such as their foundation in probability theory for combining evidence and their potential intuitive appeal. However, the proposed method has certain alternative advantages. Foremost, the method is applicable to a wide range of possible text retrieval approaches. Any approach that estimates the correct relevance ranking of documents by generating a relevance score for each document is a potential

[2] Angle between vectors is used because the fundamental calculation performed with a weight vector is inner product with the vector of expert scores. Magnitude of the weight vector is irrelevant to the overall document ranking; therefore, the only important consideration is the direction (angle) of the weight vector.

expert to be combined using the method. The approach is also sufficiently efficient to be applicable to large collections. Optimization of the 34 Megabyte CMME collection takes only a few seconds using the top-ranked 15 documents and approximately 20 minutes using all retrieved documents on a low-end Sparc IPC. In addition, the method explicitly optimizes the ability of the overall system to correctly rank documents. This is in contrast to alternatives such as the supervised learning approach examined in the previous section.

The improvements to performance demonstrated in this work are very encouraging. However, we must make clear that the amount of possible improvement depends on a number of problem-specific factors. For example, the particular experts used, the characteristics of the collection, and the quality of the training queries can all affect the performance of the optimized system. In addition, factors such as the number of documents included in training can also significantly affect the optimized performance, as the experiments demonstrated. Finally, the level of improvement may be closely tied to how representative the training queries are of typical queries to the system. If the training queries are not representative, then a system may be learned which performs well for the training queries but does not generalize well in the deployed environment.

We have demonstrated that the method can provide a novel means for analyzing the performance of experts when combined with other experts. Typical approaches for evaluating a retrieval method's performance evaluate the method in isolation. This allows for comparison between methods, but does give the system designer the information needed to decide whether a particular method should be included in the system. Echoing Belkin et al.'s [3] findings, we have demonstrated that experts which perform poorly in isolation can still contribute positively to a combined solution. Nevertheless, the issue of analyzing combinations of experts is far from resolved. First, a complete analysis must include the cost (implementation, maintenance, run-time) of including an expert; we have not considered this here. In addition, it is unclear how the best *subset* of a collection of experts can be selected, as different combinations of experts may result in different performance/cost trade-offs. These are interesting directions for future research.

7 Summary

A method for automatically finding a high performance combination of retrieval experts has been proposed. The method uses examples of past queries to learn a combination optimized to rank relevant documents before less relevant ones. The method is applicable to experts which compute ranked retrieval scores indicating the relevance of documents to queries. The method is validated by applications to two expert combination tasks. In the first task, the optimized combination of term and phrase experts performs 12% better than the best individual (the term expert), even though the phrase expert performs very poorly on its own. This illustrates the importance of evaluating experts in the context of other experts in addition to in isolation. In the second task, the optimized combination of three experts in a commercial retrieval system performs 47% better than the best individual.

Acknowledgements: This work was supported in part by NSF grant IRI-9221276. The authors thank Amy Steier for her help in developing the term/phrase experiments, and Encylopædia Britannica Inc. for access to their collections. Address all correspondence to the first author.

References

1. Brian T. Bartell. *Optimizing Ranking Functions: A Connectionist Approach to Adaptive Information Retrieval.* PhD thesis, Department of Computer Science & Engineering, The University of California, San Diego, 1994.
2. Brian T. Bartell, Garrison W. Cottrell, and Richard K. Belew. Learning the optimal parameters in a ranked retrieval system using multi-query relevance feedback. In *Proceedings of the Symposium on Document Analysis and Information Retrieval*, Las Vegas, 1994. in press.
3. Nicholas J. Belkin, C. Cool, W. Bruce Croft, and James P. Callan. Effect of multiple query representations on information retrieval system performance. In *Proc. SIGIR 1993*, pages 339–346, Pittsburgh, PA, June 1993.
4. I. Borg and J. Lingoes. *Multidimensional Similarity Structure Analysis.* Springer-Verlag, New York, 1987.
5. Edward A. Fox, M. Prabhakar Koushik, Joseph Shaw, Russell Modlin, and Durgesh Rao. Combining evidence from multiple searches. In Donna K. Harman, editor, *The First Text REtrieval Conference (TREC-1)*, pages 319–328, March 1993. NIST Special Publication 500-207.
6. Norbert Fuhr and Chris Buckley. A probabilistic learning approach for document indexing. *ACM Transactions on Information Systems*, 9(3):223–248, 1991.

7. Michael Gordon. Probabilistic and genetic algorithms in document retrieval. *Communications of the ACM*, 31(10), October 1988.

8. L. Guttman. What is not what in statistics. *The Statistician*, 26:81–107, 1978.

9. Donna Harman. Overview of the first Text REtrieval Conference. In *Proceedings of the ACM SIGIR*, pages 36–48, Pittsburgh, PA, June 1993.

10. J. Katzer, M. J. McGill, J. A. Tessier, W. Frakes, and P. DasGupta. A study of the overlap among document representations. *Information Technology: Research and Development*, 1(4):261–274, Oct 1982.

11. David D. Lewis and W. Bruce Croft. Term clustering of syntactic phrases. In *Proceedings of the ACM SIGIR*, Brussels, Sept 1990.

12. William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, 1988.

13. T. Saracevic and P. Kantor. A study of information seeking and retrieving. III. Searchers, searches, overlap. *Journal of the ASIS*, 39(3):197–216, 1988.

14. Paul Thompson. A combination of expert opinion approach to probabilistic information retrieval, part 1: The conceptual model. *Information Processing & Management*, 26(3):371–382, 1990.

15. Paul Thompson. Description of the PRC CEO algorithm for TREC. In Donna K. Harman, editor, *The First Text REtrieval Conference (TREC-1)*, pages 337–342, March 1993. NIST Special Publication 500-207.

16. Howard Turtle and W. Bruce Croft. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems*, 9(3):187–222, july 1991.

17. S. K. M. Wong, Y. J. Cai, and Y. Y. Yao. Computation of term associations by a neural network. In *Proceedings of SIGIR*, Pittsburgh, PA, June 1993.