



Available online at www.sciencedirect.com

SCIENCE @ DIRECT®

Decision Support Systems xx (2004) xxx–xxx

Decision Support
Systems

www.elsevier.com/locate/dss

On linear mixture of expert approaches to information retrieval

Weiguo Fan^a, Michael Gordon^b, Praveen Pathak^{c,*}

^aVirginia Tech, United States

^bUniversity of Michigan, United States

^c362 Stuzin Hall, PO Box 117169, Decision and Information Sciences, University of Florida, United States

Abstract

Knowledge intensive organizations have vast array of information contained in large document repositories. With the advent of E-commerce and corporate intranets/extranets, these repositories are expected to grow at a fast pace. This explosive growth has led to huge, fragmented, and unstructured document collections. Although it has become easier to collect and store information in document collections, it has become increasingly difficult to retrieve relevant information from these large document collections. Information Retrieval systems help users identify relevant documents for their information needs. Matching functions match the information in documents with that required by users in terms of queries to produce a set of documents to be presented to the users. It is well known that a single matching function does not produce the best retrieval results for all contexts (documents and queries). In this paper we combine the results obtained from well known matching functions in the literature. We employ Genetic Algorithms to do such combinations and test our method using a large well known document dataset. It is observed that our method produces better retrieval results for both the consensus search and the routing tasks in information retrieval.

© 2004 Elsevier B.V. All rights reserved.

Keywords: Information retrieval; Ranking functions; Data fusion; Genetic algorithms

1. Introduction

Although finding information on the Internet using web search engines is one of the top three Internet activities [37], experience suggests that the search engines' performance in terms of finding relevant documents for the user's query is not satisfactory. Users have to refine their search multiple times and

scan through a long list of documents only to find a few relevant documents [11]. A search engine's performance is affected by many factors like query representation, indexing, controlled vocabulary, stemming, stopping words, etc. [20]. But one of the most important factors affecting the performance is the matching function that is used to rank the documents in the collection according to their match with the user's query. These matching functions typically assign a matching score to each document in the collection and then rank the documents in the decreasing order of the score. The top few documents

* Corresponding author. Tel.: +1 352 392 9599; fax: +1 775 587 8395.

E-mail address: praveen@ufl.edu (P. Pathak).

(known as the Document Cut-off Value—DCV) are presented to the user as the result of the search. There is a variety of matching functions available in the literature in information retrieval (IR). They can be classified as:

- *Content based matching functions*
These make use of many lexical/syntactical statistics of words in a document collection e.g. token frequency (*tf*), document frequency (*df*), document length, etc. Some of the well known matching functions in this category are Okapi [31] and Pivoted TFIDF [34].
- *Link based matching functions*
These use web interconnection information to help boost the ranking performance. Some well known functions in this category are PageRank [5] and HITS [19]. These functions are quite useful to identify authoritative pages that are highly endorsed by others.
- *Structure based matching functions*
These matching functions assign weights to words appearing in different structural position of the document such as the title, and the anchor and use the weighting heuristics to improve the ranking performance.

Although a large number of matching functions have been tried in literature, no single matching function has been proved to be the best [17]. Characteristics of retrieval environment such as the size of the database, the type of the database, and the nature of the user community affects which matching function will perform better. Recently IR researchers have made efforts to combine results from multiple IR systems (using different query representations, document representations, or matching functions). This approach is the so called *data fusion* approach. In this approach results obtained by various systems on the same document database are combined. This differs from the *information fusion* approach where the results from various document databases are merged. In this paper we will use the data fusion approach. A fundamental problem in the data fusion approach is how to combine the results obtained from various IR systems. In this paper we propose a data fusion approach based on linear combinations of retrieval status values (RSV's—explained later) obtained from

different matching functions or experts¹. We use Genetic Algorithms (GA) to find the best linear combination of weights assigned to the scores of different matching functions. For comparison purposes we would also use simulated annealing approach [18] to find such weights. We will test our approach on a 10 GB web data available from TREC evaluations [12,13]. We will demonstrate how our approach can be used for both the '*Consensus Search*' task and the '*Routing*' task. Under consensus search we will evolve just one set of weights across all the user queries. Under routing task we will evolve an individual set of weights for an individual query. The importance of the consensus search task and the routing task will be explained later in the paper. Since GA's require fitness functions for training, we would like to explore the effect of fitness functions on the performance of our approach. We would use three different fitness functions to train the GA. We would compare our results against those obtained by the well known matching functions.

The paper is organized as follows. Section 2 will present related work in the area of matching function adaptations and data fusion as applied to IR. Section 3 will describe our GA based methodology in detail. We will discuss the experiments conducted in Section 4. Section 5 will present the results of our experiments and discuss the implications of the results. We will conclude the paper in Section 6 by presenting the contributions of our approach to the literature in IR and also by discussing some of the possible avenues for further research in this area.

2. Related work

Data fusion techniques in IR have typically focused on combining similarities obtained from different query representations and also on combining query representations directly [3,4]. It was found that progressive combination of different Boolean query formulations lead to improved retrieval performance as compared to performance obtained by an individual system. Combinations of the vector space model and the probabilistic models have been used [26] but

¹ We will use matching functions and experts interchangeably throughout the paper.

the results have been mixed. The fusion approach did not significantly improve the performance over that of a single system. Bartell et al. [1] used combinations of three different experts on two test collections. A ‘count’ expert counted the number of query terms in the document while two other experts analyzed different types of terms. They found that an optimized combination performed better than any individual system. Lee [21] showed how different properties of weighting schemes may retrieve different types of documents. Savoy et al. [33] combined OKAPI probabilistic model with various vector space schemes. They used a heuristic which tried to determine the best retrieval expert for a given query. Hence they did not combine any retrieval status values or document ranking information. The advantage of their approach is that the search time is limited to only one retrieval expert. But they did not find any performance gains over individual experts as determining query features that determine which retrieval expert to be used has been found to be difficult. Fox et al. [8] used six different formulae for combining similarity values of different retrieval experts. They used the minimum, maximum, or the sum of the individual similarities to combine retrieval results from different experts. They found *CombMNZ* [8] as the best performing formula. Lee [22] extended this work by analyzing the effect of using rank instead of similarity values in the functions used by Fox et al.

There have been some efforts in using linear mixture of expert approaches, which is similar in philosophy to our approach. Bartell et al. [2] have used numerical methods to optimize the parameters of a matching function. But they have chosen to optimize only the parameters involved in a standard inner product measure. Hence their adapted matching function is limited to variations of standard inner product measures. As an example, their adaptation leads to the use of one of the following matching functions: inner product, cosine, or pseudo-cosine. By contrast, our research looks at adaptation of various different forms of matching functions and is not restricted to a particular form of the matching function. In addition Bartell et al.’s work faces another limitation. The authors have assumed that the IR model have criteria (like ordering of documents) that are differentiable in nature. This assumption leads

them to use numerical methods. This assumption of existence of differentiable criteria may not hold for discrete criteria like precision and recall. Hence numerical methods may not always be useful. Our research uses genetic algorithms that do not suffer from such a limitation in assumptions regarding the nature of the criteria used.

Vogt et al. [36] used linear combinations of three experts: a binary scheme for vector space model, a tf-idf weighted scheme, and the last based on latent semantic indexing. They determined a set of parameters through training on top 100 documents for each query. Although their method worked well on training set of documents they observed that the results did not generalize well to unseen test documents. In fact their method could find results that were comparable only to the second best individual function or expert. Using our method we later demonstrate that we could get performance results which were significantly better than any of the individual matching function or expert.

In another paper Vogt and Cottrell [35] did a theoretical and empirical study on combining two IR systems and demonstrated conditions under which the linear combination model might work. However, this paper failed to provide any insights into how to combine more than two IR expert matching systems, and how to optimize the weights of combination directly on the standard performance measure like average precision. Our work does not have the limitation of just two expert systems. In fact our methodology can incorporate any number of experts in combinations. Moreover our work does not face the limitation of using any particular performance measure or the fitness function. We will demonstrate later that our method has worked well across three different well known fitness functions.

In the next section we present our methodology in details.

3. Methodology

In this section we first introduce our methodology for linear mixture of expert matching functions and then proceed to show how to use our method in real life setting involving search for new documents.

3.1. Linear mixture of expert matching functions

It is clear from previous section that combining expert approaches in information retrieval has been tried with mixed results. Either the methods were not generalizable or they assumed certain criteria which may not be always applicable in IR. Hence we propose a new method for combining expert approaches using linear combinations of retrieval status values (*RSV*) obtained by each expert matching function. We will demonstrate later that our approach does not face any of the above limitations and still beats the performance of any of the individual expert matching functions.

For any given query each expert matching function assigns a score (*RSV*) for each document. The documents in the collection are ordered in the decreasing order of *RSV* and presented to the user. We will consider four different well known experts or matching functions in our analysis. Each of them will have a *RSV* value for each document in the collection. We assign weights (in the range of -1.0 to 1.0) to the *RSV* values obtained by each expert and combine the scores to get a combined *RSV* value for each document. A negative weight attached to a *RSV* value signifies a reduced role in retrieval for the particular matching function that produced that *RSV* value. A positive weight, on the other hand, signifies an increased role in retrieval. The documents are then ordered in the decreasing order of this combined *RSV* and then presented to the user for evaluations. Mathematically the combined *RSV* can be expressed as follows:

$$\text{Combined_RSV}_m = \sum_{i=1}^n W_i S_i \quad (1)$$

Here, ' n ' is the number of expert matching functions used. S_i is the *RSV* score produced by the i th expert matching function for the m th document in the collection. W_i is the associated weight. ' i ' varies from 1 to the maximum number of experts used in the experiment. ' m ' varies from 1 to the maximum number of documents in the collection.

By proper selection of weights ' W ' it should be possible to increase the retrieval performance. This is so because the expert matching functions are com-

plementary to each other in terms of their weighting strategies for clues offered in the documents and queries. A proper selection of weights, thus, tries to exploit such complementarities. A detailed explanation about this is discussed elsewhere [28,29]. Retrieval performance is expressed in terms of *P_Avg* and *P_10* measures. These are two most important performance measures used in IR literature. *P_Avg* is the average of precisions every time a new relevant document is found, normalized by the total number of relevant documents. Here 'precision' is defined as the ratio of relevant document retrieved to the total number of documents retrieved. Recall is defined as the ratio of number of relevant documents retrieved to the total number of relevant documents in the document collection. If the exact number of relevant documents is not known in advance then a proxy for that can be used [11]. Mathematically *P_Avg* [6] can be expressed as:

$$\text{P_Avg} = \frac{\sum_{i=1}^{|D|} \left(r(d_i) * \left(\frac{\sum_{j=1}^i r(d_j)}{i} \right) \right)}{\text{TRel}} \quad (2)$$

where $r(d) \in (0,1)$ is the relevance score assigned to a document, it being 1 if the document is relevant and 0 otherwise. $|D|$ is the total number of retrieved documents. *TRel* is the total number of relevant documents for the query. It is easy to see that the *P_Avg* performance measure incorporates both the precision and recall in its calculation. It is the most widely used measure in retrieval studies for comparing performance of different systems.

P_10 performance measure is the precision obtained in the top 10 retrieved documents. This performance measure is important when the user is not interested in looking at more than just a few (in this case 10) documents presented to the user. User looking for information on the web falls under this category.

We will utilize Genetic Algorithms (GA) [15,27] to explore the search space of the weights. GA's emulate the process of the evolution of species to search for more 'fit' individuals. These algorithms are very well suited to explore complicated multidimensional

space. GA's start with a population of individuals known as *chromosomes*. Each chromosome represents a possible solution to the problem. The initial population is either randomly generated or it can also be generated using some known characteristics of the problem. The individuals in the population change with successive iterations of the algorithm (known as *generations*) following the process of *selection*, *crossover*, and *mutation*. Selection is based on the *fitness* of each chromosome in the population. Fitness is a numerical score assigned to each chromosome. It is expected that the more fit (the higher the fitness number) a chromosome the better is the utility of the chromosome in solving the problem at hand. Thus the selection of fitness function is vital for the performance of the GA. Crossover operator is used to transfer more fit building blocks from one generation to another, while the mutation operator is used to introduce random diversity in the population so that the population does not get stuck in a local optimum. The process is stopped when either the preset number of generations is reached or if there is no improvement in the performance. The chromosomes in the last generation are chosen as the best individuals to solve the problem at hand.

Our choice of GA was motivated by two factors: (a) the large size of the search space, and (b) the characteristics of the objective function. First, using the real valued expression for weights used in our model the search space is essentially infinite and the problem is basically a needle-in-a-haystack problem. Hence an exhaustive search will take inordinate time. Second, our objective function is to maximize the P_Avg value for a given query or a set of queries (routing task or consensus search task)². This is the most utilized objective function in IR studies. From Eq. (2) it is clear that the objective function is discrete in nature. GA does not require that the objective function be continuous in nature as long as it can differentiate good solutions from bad ones.

For comparison purposes we will also use simulated annealing (SA) technique [18] to search for the

weights in our model. We will report comparative results obtained by GA as well as by SA.

We now proceed to show how our method works in real life setting.

3.2. Searching for new documents

A real life search for relevant documents follows either of the two tasks: consensus search or routing [30]. In consensus search the same expert matching function is used for all types of queries issued to the system. This is the typical scenario faced by search engines where a search engine has very little knowledge about the preferences of an individual. Hence the expert matching function has to be adapted for a set of queries or for the consensus. One of the major benefits of consensus search is that it fosters result-sharing among users [30]. Contrary to this, in routing task the expert matching function is adapted to an individual query. For any new retrieval for this particular query this adapted expert matching function is used. This is the scenario for a typical knowledge worker who prefers search results to be tailored to his/her own personal preference. Most current search engines do not support such an advanced personalized search feature.

We now show how our method of evolving weights associated with expert matching functions can be used in both the consensus search and the routing task. Fig. 1 depicts how exactly the GA process will be used. In consensus search task (part *a* in figure) we will utilize just one GA process for all the queries together i.e. we will evolve just one set of weights which maximizes the retrieval performance measures for all the queries put together. In contrast, in the routing task (part *b* in figure) we will utilize the GA process for every individual query for which routing is preferred and thus evolve a personalized set of weights for the individual query.

We now show how the evolved weights in Fig. 1 can be used to retrieve new documents for either the consensus search or the routing task. The process is described in Fig. 2.

First we calculate the retrievals status values for each document for each expert matching function (S_1 , S_2 in the figure). Then we combine these scores using the weights obtained from Fig. 1. If we are looking for

² To be explained later in the paper.

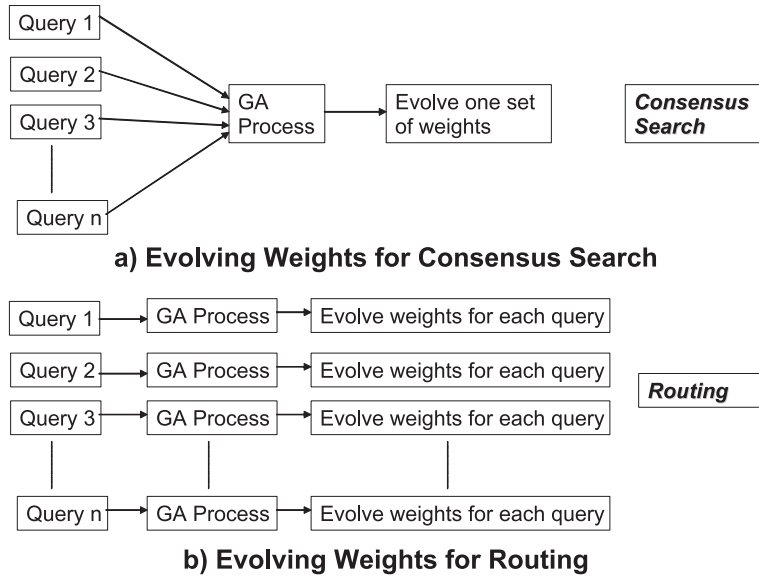
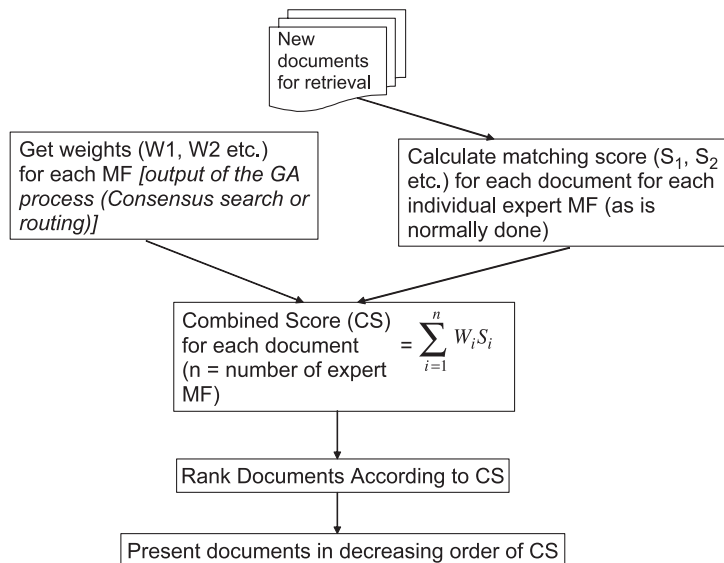


Fig. 1. Evolving weights for consensus and routing tasks.

consensus search then we use the consensus weights or else the routing weights. The combined score *CS* is used to rank the new documents in the decreasing order of the score. The top *DCV* number of documents are presented to the user. *DCV* stands for the docu-

ment cut off value which means the number of documents the user is willing to see.

The next section describes the experiments we carried out to test our methodology on a large document collection with real user queries.



MF = Matching Function

Fig. 2. New document retrieval.

4. Experiments

In this section we describe the experiment we carried out to test the viability of our methodology. First we start with description of the expert matching functions we used for combinations. Then we describe the data used in the experiments, the fitness functions used to train our GA, and finally a detailed description of the training, validation, and testing phases used during the genetic process.

4.1. Expert matching functions

Although a variety of expert matching functions have been tried in literature as mentioned in the introduction section, the content based matching functions are still quite dominant in their retrieval performance [12,13]. We decided to use four very well known content based matching functions as experts in our experiments. They are Okapi BM 25, Okapi BM2500, Pivoted TFIDF, and INQUERY [9,34]. Their choice was motivated by the fact that these experts have performed very well in the recent TREC evaluation studies. More details about these functions can be found elsewhere [9,34].

4.2. Data for experiments

We used the standard 10 GB Web Track collection from recent TREC 9 and TREC 10 conferences [12,13]. The same collection has been used extensively to evaluate various web-based information retrieval systems. Because our framework utilizes machine learning techniques, we use the residual collection method [32] to divide the entire data into three parts: training (50%), validation (20%) and test data (30%). The training data, along with the relevance information for queries is used by the GA-based system to generate a set of “candidate” weighting schemes. The validation data is used to choose the candidate scheme that has the best generalization capability for new data. The performance comparisons of all systems are based on the results on the test data only.

There are 100 topics provided in TREC 9 and 10 web track. Since 12 of these queries do not have any relevant documents in either the validation data set or the test data set, we exclude them and use the rest 88

queries as the test queries for our experiments. All of these 88 queries have their relevance information available.

DCV (document cut-off value) was set to 1000 i.e. the user is presented with top 1000 documents. This is the standard value used in TREC experiments. The fitness calculations and other performance measures are based on the top 1000 document retrieved.

4.3. Fitness functions used

To evaluate the impact of fitness functions on our methodology we use three different order-based fitness functions [23] in our experiments: P_Avg, CHK, and DCG. These have been used in other evaluation studies of fitness functions [6,24,25]. Order based fitness functions take into account not only when the relevant documents are retrieved but also the order in which they are retrieved and presented to the user. For example if we retrieve just 50 documents and only 2 of these are relevant. These relevant documents could be document number 1 and 2 in the order or they could be document 49 and 50 in the order. The order based fitness functions give greater importance to the former case when relevant documents are retrieved earlier in the order.

P_Avg is the most widely used fitness function (and performance measure) in IR. It is defined as shown in Eq. (2).

The CHK fitness function [6] is defined as follows:

$$\text{Fitness_CHK} = \frac{1}{|D|} \sum_{i=1}^{|D|} \left(r(d_i) * \sum_{j=1}^{|D|} \frac{1}{j} \right) \quad (3)$$

The notations used are same as in Eq. (2).

The DCG fitness function [16] is defined as follows:

$$\text{Fitness_DCG} = \sum_{i=1}^N \text{DCG}[i]$$

$$\text{DCG}[i] = \begin{cases} 1, & i = 1 \\ \text{DCG}[i-1] + 1/\log i, & \text{otherwise} \end{cases} \quad (4)$$

where N is the total number of retrieved documents.

4.4. Training, validation, and testing phases

The experiment is conducted in three phases. In the first step, the training data is used to train the weights associated with the chosen expert systems. The validation phase is used to choose the best weighting scheme that generalizes well on the validation data set, while in the test phase we apply these weights on the test data set. Details of the three phases are given in Fig. 3.

Training phase uses the training data. It starts with the generation of random weights associated with each of the four expert matching functions for all of the chromosomes/individuals in the population. A sample chromosome/individual in the population is as given in Table 1. A chromosome is a series of four real numbers in the range -1.0 to 1.0 . Each of the real number in the chromosome is the weight associated with an individual expert matching function.

We chose population size (the number of chromosomes in each generation) to be 75 after doing some preliminary exploratory analysis. The fitness of each individual is calculated using the chosen fitness function and the individuals are sorted in the decreasing order of their fitness values. We store the top 10 individuals for later analysis. We chose three different fitness functions (discussed later in the

Table 1

Sample chromosome				
Chromosome weights	0.3937	0.8032	-0.2927	-0.8429
Associated expert	Okapi	Okapi	Pivoted	INQUERY
matching function	BM 25	BM2500	TFIDF	

paper) to evaluate the effect of fitness functions on our methodology.

The next step is to perform genetic modifications to generate a new population. We copy the top 15% of the individuals in a generation into next generation. The remaining 85% individuals are selected using tournament selection. The crossover rate is chosen as 70%. We use Blx_alpha crossover operator as it has proved very effective in other evaluation studies with real-valued *genes* (components of chromosomes or individuals) [10,14]. In this crossover method the idea is to first get the maximum (c_{\max}) and minimum (c_{\min}) of the current parents for each of the fitness functions. Letting $I=(c_{\max}-c_{\min})$ the crossover is to randomly select a child from the $[c_{\min}-\alpha*I, c_{\max}+\alpha*I]$ where α is the crossover rate. Mutations are performed by introducing Gaussian noise in randomly selected (according to mutation rate of 15%) genes. Training was done for 30 generations. The parameters involved in genetic operations (the elitist rate, crossover rate, mutation rate, and the number of generations) were

- *Training Phase*
 - Generate initial random weights for the population
 - Repeat the following for 30 generations
 - Evaluate each individual's fitness
 - Sort the individuals in decreasing order of fitness
 - Store the top 10 individuals for later analysis
 - Perform Genetic Modification to get new population
 - Selection, Reproduction, Crossover, and Mutation
 - We have stored 300 individuals (10 times 30) at the end of training phase
- *Validation Phase*
 - Apply the top individual in each generation on the validation dataset and note the fitness
 - Choose the best performing individual to be tested on test dataset
- *Test Phase*
 - Apply the individual chosen in the validation phase on the test dataset
 - Calculate performance measures (P_Avg and P_10)
- *Calculate performance improvement over chosen expert functions*

Fig. 3. Training, validation, and test phases.

chosen after initial exploratory analysis. We carried out each step using 5 different random starting seeds. The results reported are for averages of these runs.

At the end of the training phase we have information about 300 individuals (as stated earlier we stored top 10 individuals in each of the 30 generations).

The next phase is the *validation phase*. In this phase we use the information from the training phase. For each of the 300 individuals available, Eq. (1) is used to calculate *RSV* for each document in the validation data set. The documents are arranged according to *RSV* and the fitness of each individual is calculated (with *DCV* of 1000). The best performing individual on the validation data set is used in the test phase.

The last phase of the experiment is the *test phase*. Test data set is used in this phase. We use the chosen individual from the validation phase. Eq. (1) is used to calculate *RSV* for each document in the test data set. The documents are arranged according to *RSV* and the performance measures are calculated (with *DCV* of 1000). We use the two performance measures *P_Avg* (Eq. (2)) and *P_10* discussed on pages 8 and 9 for reporting our results.

At the end of the test phase we will compare the performance results obtained by our methodology with those obtained by the standalone expert matching functions.

It is to be noted that the training and validation phases together constitute the ‘GA Process’ mentioned in Fig. 1. Thus if we are interested in the *consensus search* we would use the training and validation phase just once. In this scenario the fitness values used to train the GA will be the average of the fitness values over all the queries. At the end of the GA process we would have a set of weights that will be used (in Fig. 2) to retrieve new documents for any query. On the other hand, if we are interested in personalized search or routing then we would use the training and validation phases on each query. Thus we will have a set of weights tailored for each query. These personalized weights will be used (in Fig. 2) to retrieve new documents for that particular query.

It can be argued that the worst the GA can do is to match the performance of the best matching function as this will mean GA will assign a weight of 1.0 to the best matching function while assigning 0.0 as the

weights for the remaining matching functions. This is true for performance on training dataset as the weights are evolved on the training dataset. But the real test of our methodology is the retrieval performance on the test dataset. So it is not necessary that GA should match the performance of the best matching function for the test dataset.

For comparison purposes in addition to the GA process, we trained the weights using simulated annealing (SA) technique as well. We will report the performance measures obtained by SA and compare those with the measures obtained by the GA process.

5. Results and discussion

In this section we report the result of the experiments carried out in the last section. We first report results for the routing task in retrieval.

5.1. Results for routing task

As stated earlier, for routing task the weights have been evolved separately for every query.

Table 2 shows the performance comparisons for the routing task using *P_AVG* fitness function. Two performance measures have been reported (*P_Avg* and *P_10*) for each of the individual expert matching functions, and also for our GA based system and for simulated annealing (SA) based system. The numbers reported are the averages across all queries. It was noted that there were no significant outliers in the

Table 2
Performance comparisons for routing task with *P_AVG* fitness function

Method	Performance measures			
	<i>P_AVG</i>	Performance improvement by GA based system (%)	<i>P_10</i>	Performance improvement by GA based system (%)
GA based	0.2767		0.2966	
Linear Mixture				
Pivoted TFIDF	0.1736	59	0.2080	43
INQUERY	0.1749	58	0.2227	33
Okapi BM25	0.2247	23	0.2420	23
Okapi BM2500	0.2397	15	0.2602	14
SA based	0.2402	15	0.2613	14
Linear Mixture				

data. The table also reports the improvements in performance measures obtained by our GA based system over the other methods. We notice that our GA based system outperforms any of the individual expert matching functions on both the performance measures. In fact the performance improvements over Pivoted TFIDF, INQUERY, and Okapi BM25 are of a significantly large magnitude (ranging from 23% to 59%). Our system also outperforms the best of the individual expert matching functions, the Okapi BM2500. We conducted paired t -tests and found that the differences in performance results between our system and any of the individual expert matching functions are statistically significant (at $p < 0.05$). This is a significant result because as noted earlier some other studies [35,36] could not generalize well on unseen test document data sets with more than two expert matching functions. In fact although the earlier noted systems trained well on training data sets, they could get performance measures on the test data sets that were only as good as those obtained by the second best expert. Looking at the results for the system based on simulated annealing we note that the SA system could get results that matched the best expert matching function. Thus our approach of linear combination of retrieval status values obtained from expert matching functions can work with techniques other than GA and still get results that are at least as good as the best individual expert matching function. However, utilizing the GA-based system, we significantly outperformed even the SA based system for routing tasks. We now look at the results for the consensus search task.

5.2. Results for consensus search task

Table 3 shows the performance comparisons for the consensus search task using the P_AVG fitness function. As in Table 2 we report the P_Avg and P_10 performance measures.

The numbers reported here are the average values obtained across all the queries. As we did in routing task, in consensus search too we note that our GA based system performs significantly better than any of the individual expert matching functions, with improvements ranging from 9% to 51% in P_Avg and from 4% to 31% in the P_10 performance measure with the paired t -tests again indicating that

Table 3

Performance comparisons for consensus search with P_AVG fitness function

Method	Performance measures			
	P_AVG	Performance improvement by GA based system (%)	P_10	Performance improvement by GA based system (%)
GA based Linear Mixture	0.2617		0.2716	
Pivoted TFIDF	0.1736	51	0.2080	31
INQUERY	0.1749	50	0.2227	22
Okapi BM25	0.2247	16	0.2420	12
Okapi BM2500	0.2397	9	0.2602	4
SA based Linear Mixture	0.2387	10	0.2591	5

all the differences are statistically significant at $p < 0.05$. As in routing task, in consensus search too the SA based system could do only as well as the best individual expert matching function. The slight drop in performance for the SA based systems from the performance of the best matching function Okapi BM2500 is not statistically significant. However, the GA based system still outperformed the SA based system with statistically significant differences.

An interesting observation can be made about the drop in performance when we move from the routing task to the consensus search task. From Table 2 and Table 3 we observe that the performance of our GA based system reduces from 0.2767 to 0.2617 (5.4% drop) on P_Avg and from 0.2966 to 0.2716 (8.4% drop) on P_10 performance measures. In the routing task the weights in our system are adapted for individual queries while in consensus search task just one set of weights is evolved for the complete query set. Hence the adapted set of weights in consensus search task may not be individually optimal for individual queries. However, we see (from Table 3) that our method of evolving just one set of weights for the consensus search task still outweighs the performance of individual expert matching functions.

5.3. Effect of different matching functions

Since our GA system needs to use a fitness function for training, we wanted to test the effect of fitness function on performance. Table 4 lists the performance measures (P_Avg and P_10) obtained

Table 4
Effect of fitness functions on performance

			Fitness functions			
			P_AVG	CHK	DCG	
Task	Routing	Performance	P_Avg	0.2767	0.2747	0.2737
		measure	P_10	0.2966	0.3034	0.3068
Task	Consensus Search	Performance	P_Avg	0.2617	0.2550	0.2612
		measure	P_10	0.2716	0.2591	0.2534

while using P_Avg, CHK, and DCG fitness functions respectively. The numbers are noted for both the routing task and the consensus search task.

It can be observed from this table that the performance does not vary by much (maximum variation of 2.6%) when the performance measure is P_Avg. The variation is a little more (maximum range of 6.7%) when the P_10 performance measure is used. P_10 measure captures the information of how many relevant documents are found in the top 10 retrieved documents. Hence when we are interested in retrieving very few documents, it might be useful to select an appropriate fitness function as the choice can mean increased retrieval performance. But when we are interested in retrieving a lot of documents and using P_Avg as the performance measure, then any well known fitness function can be used.

6. Conclusion

In this paper we proposed a new genetic algorithm based data fusion approach in IR. It is based on combinations of retrieval status values obtained from different expert matching functions. The weights associated with combinations are evolved using Genetic Algorithms. We found that our method produces retrieval performance that is better than any of the individual expert matching functions used. We see a number of contributions of this research:

- Our method is independent of the individual expert matching functions used. There is no restriction on the form of the matching function. All that is required of a matching function is that it produces a retrieval status value for a document. This is a very useful property because IR researchers have developed a variety of matching

functions like structure based, content based, and link based matching functions which have completely different underlying structure. Current data fusion techniques tend to exploit this underlying structure of the matching functions which makes them restrictive in use and suitable only for certain types of matching functions. Our method can seamlessly incorporate any matching function with any underlying mechanism.

- Our method is scalable. There is no restriction on the number of expert matching functions that can be used.
- Our method works well for both the consensus search task and the routing task in information retrieval.
- Our method can incorporate any of the three well known fitness functions while evolving the weights associated with expert matching functions.

We see a number of exciting opportunities for further research in this area of research.

- Currently, we use four well-know expert matching functions in our experiment. Although these functions have very good retrieval performance, there maybe other high-performing matching functions that are yet to be discovered and later combined using our methodology. Fan et al. [6,7] have used genetic programming technique to successfully discover even better matching functions than the baseline expert matching functions used in this study. It would be interesting to see the incremental effects of combining these new ranking functions in our mixture of experts approach.
- Another direction worth exploring is the inclusion of matching functions that leverage the structural and semantic information available in HTML and XML document collection. As mentioned earlier, all four expert matching functions used in this paper are based on content information only. It would be interesting to see whether the inclusion of new types of matching functions provide any additional benefits in retrieval performance improvement.
- This paper has explored linear mixture of expert matching functions. It would be interesting to see

if a general polynomial form function can be used for combinations.

References

- [1] B. Bartell, G. Cottrell, R.K. Belew, Automatic combination of multiple ranked retrieval systems, Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR), Dublin, 1994, pp. 173–181.
- [2] B. Bartell, G. Cottrell, R.K. Belew, Optimizing similarity using multi-query relevance feedback, *Journal of the American Society for Information Science* 49 (1998) 742–761.
- [3] N. Belkin, C. Cool, W.B. Croft, J.P. Callan, The effect of multiple query representations on information retrieval performance, Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1993, pp. 339–346.
- [4] N.J. Belkin, P. Kantor, E.A. Fox, J.A. Shaw, Combining the evidence of multiple query representations for information retrieval, *Information Processing & Management* 31 (1995) 431–448.
- [5] S. Brin, L. Page, The anatomy of a large-scale hypertextual web search engine, *Computer Networks and ISDN Systems* 30 (1998) 107–117.
- [6] W. Fan, E.A. Fox, P. Pathak, H. Wu, The effects of fitness functions on genetic programming-based ranking discovery for web search, *Journal of the American Society for Information Science and Technology* 55 (2004) 628–636.
- [7] W. Fan, M. Gordon, P. Pathak, Discovery of context-specific ranking functions for effective information retrieval using genetic programming, *IEEE Transactions on Knowledge and Data Engineering* 16 (2004) 523–527.
- [8] E.A. Fox, J.A. Shaw, Combination of multiple searches, Proceedings of the 2nd Text Retrieval Conference (TREC-2), NIST, vol. 500-215, 1994, pp. 243–252.
- [9] J. Gao, G. Cao, H. He, M. Zhang, J. Nie, S. Walker, S.E. Robertson, TREC-10 web track experiments at MSRA, in: E. Voorhees, D. Harman (Eds.), Tenth text retrieval conference, NIST Special Publication, 2002, pp. 384–392.
- [10] D.E. Goldberg, Genetic algorithms in search, optimization and machine learning, Addison-Wesley, 1989.
- [11] M. Gordon, P. Pathak, Finding information on the world wide web: the retrieval effectiveness of search engines, *Information Processing & Management* 35 (1999) 141–180.
- [12] D. Hawking, Overview of the TREC-9 web track, in: E. Voorhees, D. Harman (Eds.), Ninth Text Retrieval Conference, NIST Special Publication, vol. 500-249, 2000, pp. 86–102.
- [13] D. Hawking, N. Craswell, Overview of the TREC-2001 web track, in: E. Voorhees, D.K. Harman (Eds.), Proceedings of the Tenth Text Retrieval Conference, NIST, vol. 500-250, 2001, pp. 61–67.
- [14] F. Herrera, M. Lozano, J. Verdegay, Tackling real-coded genetic algorithms: operators and tools for the behaviour analysis, *Artificial Intelligence Review* 12 (1998) 265–319.
- [15] J.H. Holland, *Adaptation in natural and artificial systems*, 2nd ed., MIT Press, 1992.
- [16] K. Jarvelin, J. Kekalainen, IR evaluation methods for retrieving highly relevant documents, Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, New York, 2001, pp. 41–48.
- [17] W.P. Jones, G.W. Furnas, Pictures of relevance: a geometric analysis of similarity measures, *Journal of the American Society for Information Science* 38 (1987) 420–442.
- [18] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, *Science* 220 (1983) 671–680.
- [19] J.M. Kleinberg, Authoritative sources in a hyperlinked environment, *Journal of the Association for Computing Machinery* 46 (1999) 604.
- [20] F.W. Lancaster, A.J. Warner, *Information retrieval today*, Information Resources Press, 1993.
- [21] J. Lee, Combining multiple evidence from different properties of weighting schemes, Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1995, pp. 180–188.
- [22] J. Lee, Analysis of multiple evidence combination, Proceedings of Twentieth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1997, pp. 267–276.
- [23] C. Lopez-Pujalte, V.P. Guerrero-Bote, F.d. Moya-Anegon, Order-based fitness functions for genetic algorithms applied to relevance feedback, *Journal of the American Society for Information Science and Technology* 54 (2003) 152.
- [24] C. Lopez-Pujalte, V.P.G. Bote, F.d.M. Anegon, A test of genetic algorithms in relevance feedback, *Information Processing & Management* 38 (2002) 793–805.
- [25] C. Lopez-Pujalte, V.P. Guerrero-Bote, F. de Moya-Anegon, Genetic algorithms in relevance feedback: a second test and new contributions, *Information Processing & Management* 39 (2003) 669–687.
- [26] C. McCabe, A. Chowdhury, D. Grossman, O. Frieder, A unified environment for fusion of information retrieval approaches, Proceedings of the Eighth International Conference on Information Knowledge Management (CIKM), Kansas City, ACM, New York, 1999, pp. 330–342.
- [27] Z. Michalewicz, *Genetic algorithms+data structures=evolution programs*, Springer-Verlag, Berlin, 1995.
- [28] P. Pathak, Use of Genetic Algorithms in Information Retrieval: Adapting Matching Functions, PhD thesis: University of Michigan, (2000) p. 141.
- [29] P. Pathak, M. Gordon, W. Fan, Effective information retrieval using genetic algorithms based matching function adaptation, Proceedings of the 33rd Hawaii International Conference on System Science (HICSS), Hawaii, USA, IEEE Computer Society Press, 2000.
- [30] J. Pitkow, H. Schutze, T. Cass, R. Cooley, et al., Personalized search, *Communications of the ACM* 45 (2002) 50.
- [31] S.E. Robertson, S. Walker, S. Jones, M.M. Hancock-Beaulieu, M. Gattford, Okapi at TREC-4, in: D.K. Harman (Ed.), Proceedings of the Fourth Text Retrieval Conference, NIST Special Publication, vol. 500-236, 1996, pp. 73–97.

- [32] G. Salton, Automatic text processing, Addison-Wesley Publishing, Reading, MA, 1989.
- [33] J. Savoy, M. Ndarugendamwo, D. Vrajitoru, Report on the TREC-4 experiment: combining probabilistic and vector space schemes, in: D.K. Harman (Ed.), Proceedings of the Fourth Text REtrieval Conference (TREC-4), NIST, 1996, pp. 537–548.
- [34] A. Singhal, G. Salton, M. Mitra, C. Buckley, Document length normalization, *Information Processing & Management* 32 (1996) 619–633.
- [35] C. Vogt, G. Cottrell, Fusion via a linear combination of scores, *Information Retrieval* 1 (1999) 151–173.
- [36] C. Vogt, G. Cottrell, R.K. Belew, B. Bartell, Using relevance to train a linear mixture of experts, in: E. Voorhees, D.K. Harman (Eds.), The Fifth Text REtrieval Conference (TREC-5), NIST, vol. 500-238, 1997, pp. 503–516.
- [37] www.searchenginewatch.com.

Weiguo Fan is an Assistant Professor of Information Systems and Computer Science at the Virginia Polytechnic Institute and State University. He received his Ph.D. in Information Systems from the Ross School of Business, University of Michigan, Ann Arbor, in 2002. His research interests include personalization, data mining, text/web mining, web computing, business intelligence, digital library, and knowledge sharing and individual learning in online communities. His research has appeared in many prestigious information technology journals such as *Information Processing and Management (IP and M)*, *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, *Information Systems (IS)*, *Decision Support Systems (DSS)*, *Journal of Management Information Systems (JMIS)*, *ACM Transactions on Internet Technology (TOIT)*, *Journal of the American Society for Information Science and Technology (JASIST)*, *Journal of Classification*, *International Journal of Electronic Business*, and in leading information technology conference such as ICIS, HICSS, AMCIS, WWW, CIKM, DS, ICOTA, etc.

Michael Gordon is Professor of Business Information Technology and Associate Dean for information technology at the Ross School of Business, University of Michigan, Ann Arbor. His research interests include information retrieval, especially adaptive methods and methods that support knowledge sharing among groups; information and communication technology in the service of social enterprise (promoting economic development, providing health care delivery, and improving educational opportunities for the poor); and using information technology along with social methods to support business education. He publishes extensively in leading IT journals such as *Information Processing and Management (IP and M)*, *IEEE Transactions on the Knowledge and Data Engineering (TKDE)*, *Decision Support Systems (DSS)*, *ACM Transactions on Internet Technology (TOIT)*, *Journal of the American Society for Information Science and Technology (JASIST)*, *Information Systems Research*, *Communication of ACM*.

Dr. Praveen Pathak is an Assistant Professor of Decision and Information Sciences at the Warrington College of Business at the University of Florida. He received his Ph.D. in Information Systems from the Ross School of Business, University of Michigan, Ann Arbor, in 2000. He also holds a MBA (PGDM) from Indian Institute of Management, Calcutta, and a Engineering degree, B. Tech. (Hons.), from the Indian Institute of Technology, Kharagpur. His research interests include information retrieval, text mining, business intelligence, and knowledge management. His research has appeared in many prestigious journals such as *Decision Support Systems (DSS)*, *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, *Journal of Management Information Systems (JMIS)*, *Information Processing & Management (IP and M)*, *Journal of the American Society for Information Science and Technology (JASIST)*, and in leading information technology conferences such as ICIS, HICSS, WITS, etc.