



Genetic algorithms in relevance feedback: a second test and new contributions

Cristina López-Pujalte ^{a,*}, Vicente P. Guerrero-Bote ^a, Félix de Moya-Anegón ^b

^a *Library and Information Science Faculty, University of Extremadura, Alcazaba de Badajoz
(Antiguo Hospital Militar), 06071 Badajoz, Spain*

^b *Library and Information Science Faculty, University of Granada, Campus Cartuja, 18071 Granada, Spain*

Accepted 25 June 2002

Abstract

The present work is the continuation of an earlier study which reviewed the literature on relevance feedback genetic techniques that follow the vector space model (the model that is most commonly used in this type of application), and implemented them so that they could be compared with each other as well as with one of the best traditional methods of relevance feedback—the Ide dec-hi method. We here carry out the comparisons on more test collections (Cranfield, CISI, Medline, and NPL), using the residual collection method for their evaluation as is recommended in this type of technique. We also add some fitness functions of our own design.

© 2002 Elsevier Ltd. All rights reserved.

Keywords: Genetic algorithms; Relevance feedback; Information retrieval; Test collections; Automatic indexing

1. Introduction

In recent years, there have appeared numerous applications of “genetic algorithms” (GAs) to information retrieval (Belew, 1989; Chen, 1995; Chen, Chung, & Ramsey, 1998; Chen & Iyer, 1998; Cordón, Moya, & Zarco, 2000, 2002; Gordon, 1988a,b, 1991; Horng & Yeh, 2000; Kraft, Petry, Buckles, & Sadasivan, 1994, 1995, 1997; López-Pujalte, 2000; López-Pujalte, Guerrero Bote, & Moya Anegón, 2002a,b; Martín-Bautista, 2000; Martín-Bautista, Vila, & Larsen, 1999; Raghavan & Agarwal, 1987; Raghavan & Birchard, 1979; Robertson & Willett, 1994, 1995, 1996;

* Corresponding author. Tel.: +34-242-89300; fax: +34-242-86401.

E-mail addresses: clopez@alcazaba.unex.es (C. López-Pujalte), vicente@alcazaba.unex.es (V.P. Guerrero-Bote), felix@goliat.ugr.es (F. de Moya-Anegón).

Sanchez, 1994; Sanchez, Miyano, & Brachet, 1995; Sanchez & Pierre, 1994; Smith & Smith, 1997; Vrajitoru, 1997, 1998; Yang & Korfhage, 1992, 1993, 1994). Most of these applications refer to “relevance feedback”, as is indicated in a review of the topic (Cordón, Moya, & Zarco, 1999).

The technique of relevance feedback is one of the most popular methods of query optimization. It consists basically in using information supplied by the user after he or she has seen a first retrieval by the system. Several studies have shown the effectiveness of the technique, which yields major improvements with respect to the original query (Salton & Buckley, 1990), so that today any information retrieval system worthy of the name has to incorporate a module that implements it.

The especial suitability of GAs to the exploration of very large dimensional vector spaces has led to their being progressively incorporated into Artificial Intelligence techniques applied to information science in general, and to information retrieval in particular, since the document spaces deriving from the application of the vector model are real spaces of very large dimensions. The algorithms lend themselves naturally to the search for improved solutions. They have also found a natural application to relevance feedback, since GAs had already been used to solve problems in which there was feedback from the environment. For instance, they had been used to fit the parameters of simulations of oil deposits, market analysis, classification, etc. (Glover, 1987; Goldberg, 1989; Grefenstette, 1986, 1987; Hilliard & Liepins, 1987; Robertson, 1987).

It is not surprising therefore that there have recently appeared many applications of GAs to relevance feedback. Most use the vector space model, which also seems to be one of the most widely used models in general (Baeza-Yates & Ribeiro-Neto, 1999). These applications implement learning of the terms and/or weights of the queries. There has been, however, no comparison of the results of the different experiments in isolation with any traditional relevance feedback algorithm.

The input to GAs (Goldberg, 1989; Holland, 1992) is a population of individuals called *chromosomes*, which represent the possible solutions to the problem. These are either generated at random or, if one has some knowledge of the problem, this knowledge can be used to create part of the initial set of potential solutions (Michalewicz, 1995). These individuals change (evolve) over successive iterations called *generations*, via processes of *selection*, *crossover*, and *mutation*. The iterations end when the system no longer improves or when a pre-set maximum number of generations has been reached. The output of the GA will be the best individual of the end population, or a combination of the best chromosomes.

For each problem to solve, one has to provide a fitness function f . Its choice is crucial for the proper functioning of the algorithm. Given a chromosome, the fitness function must return a numeric value that represents its utility. This score will be used in the process of selection of the parents, so that the fittest individuals will have a greater chance of being selected.

An application of GAs to relevance feedback may start from a set of possible solutions (the queries) which the algorithm causes to evolve with the aim of optimizing the solution. To guide the evolution of the possible solutions, one has to design fitness functions that evaluate the goodness of each solution on the basis of feedback from the user.

In previous studies that we have made to evaluate the different GAs applied to relevance feedback, applying them experimentally to the Cranfield collection (López-Pujalte et al., 2002a,b), we found that the design of the fitness function was fundamental for the GA to optimally modify the query. Indeed, it was in these functions that the different works in the literature most varied from one to another.

The intention of the present work is to bring the said studies together and amplify them to include experiments with other data collections, and thereby gain an exhaustive view of relevance feedback using genetic techniques. To this end, we used the GA that presented the best performance, running it with the different published fitness functions in the vector space model which, as we noted above, is the most commonly used model in this type of application. We applied it to four well-known test collections (*Cranfield*, *CISI*, *Medline*, and *NPL*). This allows us to generalize our earlier results and conclusions. We employed the *residual collection* method as evaluation, as recommended for this type of technique (Harman, 1992), and lastly compared the results with the classical *Ide dec-hi* feedback method, which was the method that gave the best results in the study of Salton and Buckley on the subject (1990).

2. Antecedents

We must make a special note of the work on the topic by Salton and Buckley (1990). This examines six traditionally used relevance feedback methods, and is still taken as the yardstick for investigations of the present type (Drucker, Shahrari, & Gibbon, 2002). We have taken this work as the model for our experiments, and we also compared the behaviour of all our GAs with the method that Salton and Buckley found to give the best results—the *Ide dec-hi* method (Ide, 1971).

The *Ide dec-hi* method is very simple, but nevertheless very effective. It consists in adding directly to the weights of the original query those of all the relevant documents of the set of documents provided for feedback, and subtracting from them those of the first irrelevant document obtained in the retrieval that belongs to the said set.

Formally, the query vector is reformulated as follows:

$$Q' = Q + \sum_{\text{all relevant}} D_i - S$$

where Q is the vector of the original query, D_i is the vector of the relevant document i , S is the vector of the first irrelevant document of the ranking.

With respect to works applying genetic techniques to relevance feedback in the literature (covering the period 1994–2002), most of them use the vector space model, and perform a process of learning terms and/or weights, according to the case at hand.

For obvious reasons, we cannot here give an exhaustive description of the different applications of GAs to relevance feedback. We shall only note that most of them use a very similar and basic GA, with the main difference between one and another method lying in the fitness function that is used. Hence we shall briefly comment on these different functions, which we shall later implement in an optimal feedback GA found previously (López-Pujalte, 2000), and which we shall describe in Section 3.

Yang and Korfhage (1994) propose a GA that optimizes the weights of a query by means of relevance feedback. To calculate each individual's fitness, they compare each query with the collection's documents using the Euclidean distance, retrieving those documents whose distance is less than a certain threshold. The fitness is obtained with the following expression:

$$F_1 = R_r - R_n - N_r$$

where R_r is the number of documents retrieved that belong to the standard set of relevant documents provided by the collection, R_n is the number of retrieved documents that do not belong to the standard set of relevant documents, and N_r is the number of relevant documents that were not retrieved.

From this work we obtained our functions *fitness 1* and *fitness 2*, where the latter is similar to *fitness 1*, but, instead of calculating a threshold for the retrieval, a cut-off is set for the number of documents (the first 10 documents are considered) in order to compare results with other functions that use a cut-off instead of a threshold.

Chen's group (Chen, 1995; Chen et al., 1998; Chen & Iyer, 1998) use a GA to learn the terms of a query that best represent a user-supplied set of documents (they call this process "inductive query by example"). They follow with an intelligent agent that uses this algorithm to implement the feedback module. The fitness function used by their GA consists in determining for each chromosome (possible query) the Jaccard index with the rest of the population's chromosomes. They then obtain the mean value of these similarities (our functions *fitness 3*, *fitness 4*, and *fitness 5*, with the last two being practically the same as the first but using the inner product and the cosine, respectively).

Robertson and Willett (1996) implement a GA that uses relevance feedback in learning the weights of the terms of a query. To calculate an individual's fitness they first determine the inner product of each query of the collection with each document of the database. They then retrieve a fixed number of documents, and finally calculate the recall of the retrieval. From their fitness function we will implement our functions *fitness 6* and *fitness 7*, with the latter being a variant of *fitness 6* using a cosine similarity measure instead of the inner product.

Martín-Bautista and co-workers (Martín-Bautista, 2000; Martín-Bautista et al., 1999) present an intelligent agent (GIRAF, genetic information retrieval agent filter) which can work off-line, filtering documents retrieved from Internet. It is based on a GA with fuzzy genes which carry out the adaptive learning of the user's requirements. Although these studies do not use the vector space model, in the second of them the author performs, as well as an experiment with a fuzzy representation, another experiment with a real representation, using for the fitness function the following combination of the classical measures of precision and recall:

$$F = v\text{recall} + (1 - v)\text{precision}$$

since it seems reasonable that precision should be more important than recall (in the trials the value of v is set to 0.4). This gives us our function *fitness 8*.

A most innovative work from the perspective of the fitness function is that of Horng and Yeh (2000). They use a GA to readjust the weights associated with the terms of a query in order to obtain the optimum or near-optimum query vector. The fitness function is very interesting. As well as the number of relevant and of irrelevant documents retrieved, the fitness function takes the order in which they appear into account, since it is not the same whether the relevant documents are given at the top or at the bottom of the list of retrieved documents.

Hence, it is not necessary to set a threshold that determines whether or not a document is to be retrieved. The authors consider that this characteristic is very important since they argue that the value of such a threshold affects the behaviour of the retrieval: if the value is too low, many irrelevant documents may be retrieved, and if it is too high, too few documents may be retrieved.

Their function (Chang & Hsu, 1999; Kwok, 1997) is constructed as follows: One calculates the similarity of the query vector with all the documents using the inner product, ranks the documents in order of decreasing similarity, and finally calculates the fitness of the chromosome with the following expression:

$$F = \frac{1}{|D|} \sum_{i=1}^{|D|} \left(r(d_i) \sum_{j=i}^{|D|} \frac{1}{j} \right)$$

where $|D|$ is the total number of retrieved documents, and $r(d)$ is a function giving the relevance of document d , being unity if the document is relevant, and zero otherwise. This will be our function *fitness 9*.

Finally, in López-Pujalte et al. (2002b), we worked with two fitness functions also based on the order of appearance of the documents. For the first of the functions (which will be our *fitness 10*), we use an additional parameter A which determines the value of the factors used in the function. The function also performs a comparison of the indicated chromosome and the feedback documents, using the cosine as similarity measure. After ranking the documents by relevance, we make a cumulative calculation for all the retrieved documents in the following form:

$$T = \sum_{\text{pos}=1}^{\text{num}} \left(r(d_i) \frac{1}{A} \left(\frac{A-1}{A} \right)^{\text{pos}-1} \right)$$

where pos is the position of the document under consideration, num the total number of documents retrieved, and $r(d)$ a function returning the relevance of document d , being $+1$ if the document is relevant, and -1 otherwise. The contribution will therefore be positive for a relevant document and negative for an irrelevant document. The optimal value of the parameter A was fixed by experiment to 2.0.

The accumulated total result is finally multiplied by the recall of the retrieval to give the chromosome's fitness.

One sees that, like other order-based fitness functions, this function needs neither a threshold nor a fixed cut-off of the number of documents. Instead it considers all the documents that have been retrieved, i.e., with similarity greater than zero.

The second of our functions (*fitness 11*) consists basically of finding the mean precision in nine recall intervals (0.1, 0.2, ..., 0.9), including an interpolation procedure to remove ambiguities.

First, the cosine method is used to calculate the similarity between the indicated chromosome and the feedback documents. The documents are then ranked into decreasing degree of similarity, and the mean precision is calculated in the nine recall intervals basically by assigning as the precision of an interval the highest precision of all the recall points that rank above it to the right. This mean precision is the fitness value of the given chromosome.

3. The genetic algorithm

As we mentioned above, in the present work we use a GA implemented for relevance feedback and optimized in earlier work (López-Pujalte, 2000) in which we experimented with the alternatives or improvements to the classical model taken from the literature and some of our own

design (López-Pujalte et al., 2002a). In the following, we shall give some details of the characteristics of the GA that gave the best performance, and on which we shall implement each of the fitness functions defined in the previous section. These functions will guide the algorithm in its search process.

3.1. Representation of the chromosomes

The vectors corresponding to the documents provided as feedback are subjected to a conversion process such as that used by Chen (1995) to transform them into the chromosomes that our GA will work with. These chromosomes use a real representation, and will have the same number of genes (components) as the query and the feedback documents have terms with non-zero weights. First, the set of terms contained in those documents and the query is calculated, and the size of the chromosomes will be equal to the number of terms of that set.

3.2. The population

Our GA receives an initial population consisting of the chromosomes corresponding to the relevant documents, the irrelevant documents, the latter with their terms negated (of course, in all cases we are referring to the feedback documents), and lastly to the supposedly optimized query. This is, therefore, a hybrid GA (Davis, 1991).

3.3. Selection

As the selection mechanism, the GA uses “simple random sampling” (Goldberg, 1989; Holland, 1992). This consists in constructing a roulette with the same number of slots as there are individuals in the population, and in which the size of each slot is directly related to the individual’s fitness value. Hence, the best chromosomes will on average achieve more copies, and the worst fewer copies. Our GA also uses the “elitism” strategy (De Jong, 1975), as a complement to the selection mechanism. If, after generating the new population, the best chromosome of the preceding generation is by chance absent, the worst individual of the new population is withdrawn and replaced by that chromosome.

3.4. Genetic operators

Our algorithm uses the *one-point crossover* operator and *random mutation* (Goldberg, 1989; Holland, 1992; Michalewicz, 1995). We also found that the GA’s behaviour was improved by including a process of normalization of all the population’s chromosomes each time the genetic operators were applied to them (even though they came from already normalized documents).

3.5. Control parameters

All the control parameters were determined experimentally. We carried out numerous trials in order to obtain optimal values of these parameters. The values of the control parameters crossover probability p_c and mutation probability p_m with which we obtained the best results were

markedly higher than those normally used, especially ($p_c = 0.8$ and $p_m = 0.2$), since this helps to maintain the population's diversity and avoid premature convergence of the algorithm. We performed trials with 10, 20, 40, 80, 100, and 200 generations, and finally chose 20 since from then on the population showed no further improvement.

3.6. Solution

On termination, the GA returns as the solution the best chromosome found during the process, as well as the *centroid* of those chromosomes of the end population which have a fitness equal to the greatest value found in that generation. This second solution reflects the idea of not wasting potentially valuable information, since if there was more than one chromosome in the last generation with the maximum fitness, what criterion should one use to select one and reject the others? We therefore calculate the centroid, i.e., the vector (chromosome) whose components are the arithmetic means of the components of those vectors (chromosomes) whose fitness is equal to the maximum value found in the last generation. Formally, the centroid is calculated according to the following expression:

$$t_{iR} = \frac{\sum_{j=1}^T f(\max) C_{ij}}{\sum_{j=1}^T f(\max)}$$

where t_{iR} is the term i of the resulting chromosome, T is the total size of the population, $f(\max)$ is a function which returns 1 if the fitness of the chromosome is equal to the maximum fitness of the population, and 0 otherwise, C_{ij} is term i of chromosome j .

We implemented and evaluated both solutions.

4. Experiment and evaluation

To perform the trial, it was first necessary to generate test databases. We created these from four of the best known test collections: the Cranfield collection (1398 documents on different aspects of aeronautical engineering, and 225 queries for which the relevance scores are known), the CISI collection (1460 documents on information science and 76 queries), Medline (1033 documents on medicine and 30 queries), and lastly the NPL collection (11,429 documents and 93 queries on electronic engineering).

One of the principal reasons for the choice of these collections was that they had been used elsewhere for feedback experiments, and were also used in the work of Salton and Buckley which we take as our benchmark with which to compare the results. Moreover, these four collections present very different characteristics (see Tables 1 and 2) in subject matter, record structure, size, mean lengths of documents and queries (expressed in number of terms), mean number of relevant documents per query, etc. Most of these characteristics have a major influence on the relevance feedback process, so that with these four collections we would have a diverse and fairly complete testbed.

Since relevance feedback is the technique that is the object of the present work, it was necessary to analyse the example queries that come with each collection, in order to select those which were suitable for testing that technique.

Table 1
Collection statistics (adapted from Salton & Buckley, 1990)

Collection	No. of vectors	Mean length of vector	Standard deviation of length of vector	Mean frequency of the terms in the vectors	Percentage of terms in vectors with frequency 1
<i>Cranfield</i>					
Documents	1398	53.13	22.53	1.58	69.50
Queries	225	9.17	3.19	1.04	95.69
<i>CISI</i>					
Documents	1460	46.55	19.38	1.37	80.27
Queries	76	28.29	9.49	1.38	78.36
<i>Medline</i>					
Documents	1033	51.60	22.78	1.54	72.70
Queries	30	10.10	6.03	1.12	90.76
<i>NPL</i>					
Documents	11,429	19.96	10.84	1.21	84.03
Queries	93	7.16	2.36	1.00	100.00

Table 2
Means related to the relevant documents of the collections, and queries of each of them appropriate to the relevance feedback technique

Collections	Mean relevant documents	Mean retrieved relevant documents (with cut-off of 15)	Queries used in the trials
Cranfield	7.04	2.28	33
CISI	40.96	3.69	37
Medline	23.16	7.46	27
NPL	22.39	2.55	32

We initially set 15 as being the number of documents to implement the feedback, as was done in the experiments of Salton and Buckley (1990). That is, for each query the first 15 documents retrieved were examined to determine their relevance, and this information is what was given to the algorithm as feedback.

Not all the queries in the collections were suitable to test the technique. Queries which retrieve all their relevant documents amongst the first 15 are not suitable since there are no documents left which are interesting to retrieve, and hence they are no use in measuring the results. Likewise, neither were queries that retrieved no relevant documents amongst these 15 suitable, since they can therefore provide no information for feedback.

Hence, we selected around 30 queries for each collection that gave at least three relevant documents amongst the first 15 retrieved, and lacked at least five relevant documents still not retrieved. The experimental trials were carried out on these subsets of each collection's example queries (Table 2).

4.1. Prior phase: document vectorization

Before continuing with the experiment, it was necessary to convert the documents and queries of each collection to the form of document vectors. Firstly, to determine the terms that we would

use to describe the documents, we used the following procedure which we had also used in earlier experiments (Guerrero Bote & Moya Anegón, 2001; Guerrero Bote, Moya Anegón, & Herrero Solana, 2002):

1. Extraction of all the words from each document.
2. Elimination of the stopwords from a stopword list generated with the frequency dictionary of Kucera and Francis (1967).
3. Stemming the remaining words using the *Porter Stemmer* which is the most commonly used stemmer in English (Frakes & Baeza-Yates, 1992; Porter, 1980).

After this procedure, the final number of terms was 4307 for the Cranfield collection (so that we would be working with 1398 document vectors of 4307 components), 5572 for the CISI collection (1460 5772-component vectors), 8886 for Medline (1033 8886-component vectors), and 7891 for NPL (11,429 7891-component vectors). We have to note that there were memory problems with this last collection.

Having thus determined the terms that described all the documents of the collection, we next assigned the weights using the same scheme proposed by Salton and Buckley (1990), in order to equiparate in so far as possible our experiment with theirs. The expression is:

$$a_{ij} = \frac{\left(0.5 + 0.5 \frac{tf_{ij}}{\max tf}\right) \log \frac{N}{n_i}}{\sqrt{\left(0.5 + 0.5 \frac{tf_{ij}}{\max tf}\right)^2 \left(\log \frac{N}{n_i}\right)^2}}$$

where a_{ij} is the weight assigned to the term t_j in document D_i , tf_{ij} is the number of times that term t_j appears in document D_i , n_j is the number of documents indexed by the term t_j , and lastly N is the total number of documents in the database.

Lastly, we normalized the vectors, dividing them by their Euclidean norm, since according to the study of Noreault, McGill, and Koll (1981), the best similarity measures are those which make angle comparisons between vectors.

We carried out a similar procedure with the collections' associated queries, thereby obtaining the normalized query vectors which will be the object of relevance feedback optimization.

4.2. Design of the experiment

The experimental scheme with which to implement relevance feedback using the different methods (the GAs and the Ide dec-hi method) is very simple (López-Pujalte, 2000):

- For each collection, each query is compared with all the documents, using the cosine similarity measure. This yields a list giving the similarities of each query with all the documents of the collection.
- This list is ranked in decreasing order of degree of similarity.
- The normalized document vectors corresponding to the top 15 documents of the list (which will be those to use as feedback), with their relevance scores and the normalized query vector, are provided as input to the query optimization algorithm.

- The program also outputs a masking file which contains for each query all the documents that must not be considered in the evaluation process (these will be the first 15 which have been used in modifying the queries). This is the residual collection method, as used by Salton and Buckley (1990).

4.3. Evaluation

We evaluated the results of the retrieval via the classical measures of *recall* and *precision*. The latter was calculated by interpolation at fixed recall intervals (Salton & McGill, 1983), finding the average precision for three recall values (0.25, 0.5, and 0.75, representing low, medium, and high recall, respectively) as was done by Salton and Buckley (1990) in their study of feedback so as to be able to compare the different systems.

We also used the residual collection method (Chang, Cirillo, & Razon, 1971), in which all the documents previously seen by the user (whether relevant or not) are removed from the collection, and both the initial and the modified query are evaluated on this residual collection, i.e., they are evaluated on the collection that results from taking all of the documents used as feedback out of the original collection. The idea is that the relevance feedback operation has to be judged on the basis of its performance in retrieving new documents as yet unexamined by the user. This is the standard relevance feedback evaluation method, since it provides a more realistic and impartial assessment (Baeza-Yates & Ribeiro-Neto, 1999; Harman, 1992; Salton & Buckley, 1990). It has to be borne in mind however that this method maintains a correct difference between the initial execution and the executions with feedback at the cost of precision and recall. The results from this procedure cannot therefore be directly compared with those obtained by a classical method of evaluation that does not use the residual collection.

5. Results

In this section we shall present the results of the trials, classified according to test collection, and finally comment on the results overall. Before beginning, regarding the form of calculation of the solution employed by the GA, we noted above that there were two alternatives, reflected in the tables of results (Tables 3–6):

- The “best” solution: this is the best chromosome found at any point in the process.
- The “centroid” solution: this is the centroid of the chromosomes of the end population whose fitness value is the maximum value calculated for that population.

The tables of results give the mean precision calculated at fixed recall intervals as was described in the previous section for both the Ide dec-hi method and the different GAs in which case the table lists the means given by the two methods of obtaining the solution. The percentage improvement of each relevance feedback technique is calculated relative to the mean obtained for the queries without using feedback (our baseline), for both the Ide dec-hi method and the different GAs.

Table 3
Results of different relevance feedback methods for the Cranfield collection

Method	Best		Centroid	
	Mean	Improve (%)	Mean	Improve (%)
No feedback	0.098			
Ide (dec-hi)	0.218	120.8		
<i>GAs</i>				
GA with fitness 1	0.148	50.7	0.120	22.4
GA with fitness 2	0.218	120.8	0.150	53
GA with fitness 3	0.043	–	0.043	–
GA with fitness 4	0.060	–	0.030	–
GA with fitness 5	0.053	–	0.053	–
GA with fitness 6	0.218	120.8	0.151	54
GA with fitness 7	0.218	120.8	0.154	57.1
GA with fitness 8	0.218	120.8	0.151	54
GA with fitness 9	0.218	120.8	0.220	123.6
GA with fitness 10	0.218	120.8	0.210	114.2
GA with fitness 11	0.218	120.8	0.224	127.2

Table 4
Results of different relevance feedback methods for the CISI collection

Method	Best		Centroid	
	Mean	Improve (%)	Mean	Improve (%)
No feedback	0.129			
Ide (dec-hi)	0.144	11.5		
<i>GAs</i>				
GA with fitness 1	0.123	–	0.105	–
GA with fitness 2	0.144	11.5	0.121	–
GA with fitness 3	0.059	–	0.061	–
GA with fitness 4	0.056	–	0.056	–
GA with fitness 5	0.075	–	0.068	–
GA with fitness 6	0.144	11.5	0.121	–
GA with fitness 7	0.142	9.9	0.123	–
GA with fitness 8	0.144	11.5	0.121	–
GA with fitness 9	0.140	8.2	0.142	10
GA with fitness 10	0.132	2.3	0.105	–
GA with fitness 11	0.144	11.5	0.138	7.4

5.1. Results on the Cranfield collection

For the Cranfield collection (Table 3), the results obtained by the GAs varied widely depending on which fitness function was used—from no improvement with the fitness functions introduced by Chen's group and their variants, to an improvement of 127% with one of the functions that we had designed, fitness function 11.

Table 5
Results of different relevance feedback methods for the Medline test collection

Method	Best		Centroid	
	Mean	Improve (%)	Mean	Improve (%)
No feedback	0.458			
Ide (dec-hi)	0.730	59.5		
<i>GAs</i>				
GA with fitness 1	0.477	4.1	0.401	–
GA with fitness 2	0.730	59.5	0.635	38.7
GA with fitness 3	0.434	–	0.339	–
GA with fitness 4	0.363	–	0.224	–
GA with fitness 5	0.403	–	0.281	–
GA with fitness 6	0.730	59.5	0.635	38.6
GA with fitness 7	0.718	56.8	0.631	37.8
GA with fitness 8	0.730	59.5	0.637	39
GA with fitness 9	0.714	56	0.697	52.3
GA with fitness 10	0.651	42.2	0.428	–
GA with fitness 11	0.730	59.5	0.722	57.7

Table 6
Results of three relevance feedback methods for the NPL collection

Method	Best		Centroid	
	Mean	Improve (%)	Mean	Improve
Without feedback	0.130			
Ide (dec-hi)	0.150	15.4		
<i>Genetic</i>				
GA with fitness 6	0.134	3.7	0.109	–
GA with fitness 9	0.107	–	0.122	–
GA with fitness 11	0.145	12.1	0.119	–

One sees therefore that the relevance feedback technique which yields the best results is the genetic technique using fitness function 11, followed by another genetic technique using fitness function 9 designed by Horng and Yeh (2000). Both functions take the order of appearance of the relevant documents into account.

One sees from Table 3 that in most cases the GA gives better results with the first method (the best chromosome), except for the GAs that use fitness functions 9 and 11 which give better results with the second method (the centroid).

The comparison with the traditional Ide dec-hi method showed that, on this collection, most of the GAs that were implemented matched the great improvement obtained by the traditional method (120.8%), and two of them, those which used functions 9 and 11, even surpassed that improvement (one of them by a further 7%).

5.2. Results on the CISI collection

In the case of the CISI collection (Table 4), the improvement obtained with feedback is far less than was obtained with the Cranfield collection (now around 10% for those methods which function positively with this technique, as against 120% for the Cranfield collection). This is because of the particular characteristics of the two collections, since one finds that the collections with short queries (Cranfield, for instance, with a mean length of its queries of 9.2 terms) tend to gain more in the feedback procedure than those collection which have large queries, such as in the present case (mean query length of 28.3 terms). This behaviour is also found with the traditional methods of relevance feedback (Salton & Buckley, 1990).

Also, the GAs which have a poor behaviour in this collection are the same as in the Cranfield collection, but with a new one corresponding to fitness function 1. This had obtained an improvement of 50.7% with the Cranfield collection. It was introduced by Yang and Korfhage (1994), who in that work only tested it on the Cranfield collection. The reason for the marked difference may be that the threshold is not appropriate for the CISI collection, since the same function implemented with a cut-off for the number of documents (fitness 2) instead of a threshold yielded good results.

Otherwise, there is a great degree of uniformity amongst the remaining GAs. The best are fitness 11 (as also with the Cranfield collection), and fitnesses 2, 6, and 8, all with improvements of around 11%. These are followed by fitness 9 with 10% improvement, and only fitness 10 gave a very small improvement (2.3%).

With respect to the two methods used to calculate the solution, most of the GAs gave better results with the first method (Table 4), except the GA using fitness 9 which gave better results using the centroid of the chromosomes of the end population whose fitness value was the same as the maximum calculated for that population.

As we mentioned above, the characteristics of this collection led to the improvement being notably less than on the Cranfield collection. The same is the case with traditional techniques as with the GAs, since the *Ide dec-hi* method here only achieves an 11.5% improvement, the same as the GAs. Comparison of the two techniques (genetic and traditional) applied to this collection, only a third of the GAs that were implemented attained the improvement reached by the traditional method (although the others came fairly close).

5.3. Results on the Medline test collection

With the Medline test collection (Table 5), the improvement obtained with relevance feedback was notable, being almost 60% in the methods using feedback, although this is nowhere near the values attained with this technique on the Cranfield collection. As we remarked above, the fact that this is also a collection with small query examples, although somewhat larger than those of the Cranfield collection (mean query length of 10.10 terms), may have had an influence.

There is still uniformity with respect to which GAs behave poorly and which well in the different collections. In this collection, however, all the GAs gave better results using the first method to generate the solution.

Lastly, we must note that, with this collection, the Ide dec-hi method gave a 59.5% improvement, and that, as was the case above with the CISI collection, a third of the GAs (exactly the same algorithms as with the previous collection) achieved this improvement.

5.4. *Results on the NPL collection*

The NPL collection was a case apart. It did not follow the patterns more or less established in the other collections. Indeed, this is a somewhat odd collection that has been earning itself a reputation in being a misfit in experiments where it has been used. Our case it seems was going to be no exception. It also behaved differently from the rest in the work of Salton and Buckley (1990) on traditional feedback techniques.

The reason is doubtless that the characteristics of this collection are substantially different from the rest of the collections (Salton & Buckley, 1988): as well as the smallness of its documents and query examples (they are the smallest of all the collections, with mean lengths of 19.96 terms per document and 7.16 terms per query), the frequencies of the terms are especially low. Each term of a query appears only once, and the mean frequency of the terms in the documents is the lowest (1.21). Under these circumstances, the frequency-based weight of terms and the normalization operations are unable to properly fulfil their function.

It has also been conjectured that the NPL indexing terms were carefully chosen, and may in fact represent controlled terms rather than freely chosen natural language inputs (Salton & Buckley, 1990). This is not the case with the other collections.

As we noted above, the memory problems with this collection were real headaches for us because of the enormous number of vectors of great size (11,429 7891-component document vectors). We therefore only made the pertinent adaptations and ran three of the GAs that had shown good behaviour with the rest of the collections. In no case did the improvement equal that obtained with the classical Ide dec-hi method.

The results on this collection are given in Table 6. One sees that, of the GAs tested, the only one that approached the Ide dec-hi method improvement was that using fitness 11 of our own design.

With respect to the method followed to obtain the solution, the second (centroid) method was better for the GA using fitness 9, while the first was better for the GAs that used fitnesses 6 and 11.

With this collection, the best results were obtained with the Ide dec-hi method, the improvement being 15.4%.

5.5. *Discussion of the results*

In sum, we can conclude that the relevance feedback techniques improve the behaviour of an information retrieval system by percentages ranging from the 11.5% obtained on the CISI collection, to the 127% on the Cranfield collection, in a single iteration of the feedback process. The results depend firstly on the specific characteristics of the document collection being dealt with, and then, in the case of the GA-based techniques, on the fitness function used.

Of the four test collections tested, the best behaviour (with both the Ide dec-hi method and the genetic techniques) was the Cranfield, and indeed it was on this collection that the genetic method surpassed the improvement of the classical method by nearly 7%. Given the influence of the choice of fitness function on the performance of a GA, it has to be borne in mind that these results can

surely be further improved, since one could continue experimenting to try to achieve a really optimal fitness function. Also, one could implement selection mechanisms, and mutation and crossover operators that are more sophisticated than those to be found in the literature as of now, and that would probably improve the performance of these algorithms.

Leaving aside the NPL collection, which, as we noted above, cannot be directly compared with the rest of the collections because of its idiosyncracies, the Cranfield test collection is followed in the improvements obtained with both the traditional and genetic methods by the Medline and the CISI test collections, in that order. This was also the finding of the work of Salton and Buckley (1990) with the traditional relevance feedback techniques.

6. Conclusions

The document spaces derived from application of the vector space model are spaces of large dimensions (in the present work, for example, for one of the test collections we have 11,429 vectors each of 7891 components). Since genetic algorithms have a proven efficacy in exploring large complex spaces, they may be expected to give good results in exploring document spaces. There would therefore seem to be a major field of application of GAs to information retrieval.

Furthermore, given the relative ease of implementation of relevance feedback (whether in genetic or more traditional methods) and the excellent results being achieved with it, any information retrieval system worthy of its name should today incorporate a feedback module.

It was in the fitness functions, which allow the goodness of each of the possible solutions to be evaluated, that the experiments that have been carried out on this topic most varied from one to another. The choice of fitness function is of prime importance for good improvements to be achieved, since we have seen that some functions can lead to success and others to utter failure in the exploration.

In the present work, we have tested the different functions present in the literature that have been applied within the vector space model, and two other fitness functions of our own design. To test the GAs and the different functions that were implemented, we used four different test collections. On the one hand, this meant that we did not ourselves establish the relevance judgments, and on the other that the variety of the collections gave results in different settings, with documents of diverse characteristics, and which are therefore more reliable and generalizable.

One can deduce from the results that these GAs allow one to obtain a considerable improvement of the original query in a single iteration (at times the improvement was found to be 127%). These results equal, or even in some cases surpass, the best of the classical methods used in relevance feedback—the *Ide dec-hi* method—according to the work of Salton and Buckley (1990). The GA surpasses the classical method by almost 7%. While this improvement is small, it was nonetheless found to be statistically significant under both Student's *t*-test and the sign test.

These improvements obtained with feedback depend firstly on the specific characteristics of the document collection being dealt with, and, in the case of GA-based techniques, on the fitness function that is used. It must be noted that there were no differences between the classical and the genetic methods of implementing the relevance feedback technique in their behaviour on the different collections. In both cases, the Cranfield test collection responded best, followed by

Medline and CISI. The NPL collection was, as expected, a case apart—it did not follow the patterns established in the other collections.

Clearly, the characteristics of the collections have a determining influence on the feedback process. For instance, one of the influencing factors is the mean length of the queries, since the collections with short (often incomplete) queries tended to gain more than the collections with longer and more varied queries. Another factor seems to be the proportion of relevant documents that are retrieved amongst the documents that will make up the feedback (in our case, the first 15 documents). Hence, the collections with the greater percentages are those that respond better to relevance feedback, since, naturally, that feedback information will be more useful and valuable.

With respect to the fitness functions, one can also conclude that, of the functions tested, the best results were given by those that take into account not just which documents are retrieved, but also the order in which they are retrieved. That is, these are fitness functions which value not only that the possible solution retrieves many relevant documents and few irrelevant documents, but also whether the relevant documents are at the top or the bottom of the list.

The fitness function that gave the best results was one that we had designed. It simulates a retrieval process among the feedback documents, and calculates the mean precision in nine recall intervals from 0.1 to 0.9.

As immediate lines of further study, we are thinking of experimenting with modifications of the GA that include, above all, selection mechanisms and crossover operators that are more appropriate for a real representation, of course without leaving the line of investigating the design of fitness functions, since we believe that there is where a most important key to the performance of our GA lies.

Acknowledgements

This work was financed by the Junta de Extremadura-Consejería de Educación Ciencia & Tecnología and the Fondo Social Europeo, as part of research project IPR99A047.

References

- Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. Essex, UK: Addison-Wesley.
- Belew, R. K. (1989). Adaptive information retrieval. In *Proceedings of the association for computing machinery special interest group on information retrieval (ACM/SIGIR) 12th annual international conference on research and development in information retrieval, 25–28 June 1989, Cambridge, MA* (pp. 11–20). Nueva York, NY: ACM, Inc., ISBN 0-89791-321-3.
- Chang, C.-H., & Hsu, C.-C. (1999). *The design of an information system for hypertext retrieval and automatic discovery on WWW*. Ph.D. Thesis, Department of CSIE, National Taiwan University.
- Chang, Y. K., Cirillo, C., & Razon, J. (1971). Evaluation of feedback retrieval using modified freezing, residual collection, and test and control groups. In G. Salton (Ed.), *The smart retrieval system-experiments in automatic document processing* (pp. 355–370). Englewood Cliffs, NJ: Prentice-Hall.
- Chen, H. (1995). Machine learning for information retrieval: neural networks, symbolic learning, and genetic algorithms. *Journal of the American Society for Information Science*, 46(3), 194–216.

- Chen, H., Chung, Y., & Ramsey, M. (1998). A smart itty bitsy spider for the web. *Journal of the American Society for Information Science*, 49(7), 604–618.
- Chen, H., & Iyer, A. (1998). A machine learning approach to inductive query by examples: an experiment using relevance feedback, ID3, genetic algorithms, and simulated annealing. *Journal of the American Society for Information Science*, 49(8), 693–705.
- Cordón, O., Moya, F., & Zarco, M. C. (1999). Breve estudio sobre la aplicación de los algoritmos genéticos a la recuperación de la información. In *IV Congreso ISKO* (pp. 179–186). London: Granada.
- Cordón, O., Moya, F., & Zarco, M. C. (2000). A GA-P algorithm to automatically formulate extended boolean queries for a fuzzy information retrieval system by means of GA-P techniques. *Mathware & Soft Computing*, 7(2–3), 309–322.
- Cordón, O., Moya, F., & Zarco, M. C. (2002). A new evolutionary algorithm combining simulated annealing and genetic programming for relevance feedback in fuzzy information retrieval systems. *Soft Computing*, in press.
- Davis, L. (Ed.). (1991). *Handbook of genetic algorithms*. New York: Van Nostrand Reinhold.
- De Jong, K. A. (1975). *An analysis of the behavior of a class of genetic adaptive systems*. Ph.D. Thesis, University of Michigan.
- Drucker, H., Shahrany, B., & Gibbon, D. C. (2002). Support vector machines: relevance feedback and information retrieval. *Information Processing and Management*, 38, 305–323.
- Frakes, W. B., & Baeza-Yates, R. (1992). In W. B. Frakes & B. Y. Ricardo (Eds.), *Information retrieval: data structures & algorithms*. Englewood Cliffs, NJ: Prentice-Hall.
- Glover, D. E. (1987). Solving a complex keyboard configuration problem through a generalized adaptive search. In L. Davis (Ed.), *Genetic algorithms and simulated annealing* (pp. 12–31). San Mateo, CA: Morgan Kaufmann.
- Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*. Reading, MA: Addison-Wesley.
- Gordon, M. D. (1988a). Probabilistic and genetic algorithms for document retrieval. *Communications of ACM*, 31(10), 1208–1218.
- Gordon, M. D. (1988b). The necessity for adaptation in modified Boolean document retrieval systems. *Information Processing and Management*, 24(3), 339–347.
- Gordon, M. D. (1991). User-based document clustering by redescribing subject descriptors with a genetic algorithms. *Journal of the American Society for Information Science*, 42(5), 311–322.
- Grefenstette, J. J. (1986). Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics, SMC*, 16(1), 122–128.
- Grefenstette, J. J. (1987). Incorporating problem specific knowledge into genetic algorithms. In L. Davis (Ed.), *Genetic algorithms and simulated annealing* (pp. 42–60). Los Altos: Morgan Kaufmann Publishers.
- Guerrero Bote, V. P., & Moya Anegón, F. (2001). Reduction of the dimension of a document space using the fuzzified output of a Kohonen network. *Journal of the American Society for Information Science and Technology*, 52, 1234–1241.
- Guerrero Bote, V. P., Moya Anegón, F., & Herrero Solana, V. (2002). Document organization using Kohonen's algorithm. *Information Processing and Management*, 38, 79–89.
- Harman, D. K. (1992). Relevance feedback and other query modification techniques. In W. B. Frakes & R. Baeza-Yates (Eds.), *Information retrieval: data structures and algorithms* (pp. 241–263). Englewood Cliffs, NJ: Prentice-Hall.
- Hilliard, M. R., & Liepins, G. E. (1987). A classifier-based system for discovering scheduling heuristics. In *Genetic algorithms and their applications: proceedings of the second international conference on genetic algorithms* (pp. 231–235).
- Holland, J. H. (1992). *Adaptation in natural and artificial systems*. (2nd ed.). Cambridge, MA: MIT Press.
- Hornig, J.-T., & Yeh, C.-C. (2000). Applying genetic algorithms to query optimization in document retrieval. *Information processing and management*, 36, 737–759.
- Ide, E. (1971). New experiments in relevance feedback. In G. Salton (Ed.), *The SMART retrieval system* (pp. 337–354). Englewood Cliffs, NJ: Prentice-Hall.
- Kraft, D. H., Petry, F. E., Buckles, B. P., & Sadasivan, T. (1994). The use of genetic programming to build queries for information retrieval. In *Proceedings of IEEE symposium on evolutionary computation*. Orlando, FL.

- Kraft, D. H., Petry, F. E., Buckles, B. P., & Sadasivan, T. (1995). Applying genetic algorithms to information retrieval systems via relevance feedback. In P. Bosc & J. Kacprzyk (Eds.), *Fuzzy sets and possibility theory in database management systems, studies in fuzziness series* (pp. 330–346).
- Kraft, D. H., Petry, F. E., Buckles, B. P., & Sadasivan, T. (1997). Genetic algorithms for query optimization in information retrieval: relevance feedback. In E. Sanchez, T. Shibata, & L. A. Zadeh (Eds.), *Genetic algorithms and fuzzy logic systems. Soft computing perspectives* (pp. 155–173).
- Kucera, H., & Francis, N. (1967). *Computational analysis of present-day American English*. Providence, RD: Brown University Press.
- Kwok, K. L. (1997). *Comparing representations in Chinese information retrieval*. Philadelphia, PA, USA: ACM/SIGIR, p. 34–41.
- López-Pujalte, C. (2000). *Algoritmos genéticos aplicados a la retroalimentación por relevancia*. Ph.D. Thesis, Library and Information Science Faculty, University of Granada.
- López-Pujalte, C., Guerrero Bote, V. P., & Moya Anegón, F. (2002a). A test of genetic algorithms in relevance feedback. *Information Processing and Management*, in press.
- López-Pujalte, C., Guerrero Bote, V. P., & Moya Anegón, F. (2002b). Order-based fitness functions for genetic algorithms applied to relevance feedback. *Journal of the American Society for Information Science and Technology*, in press.
- Martín-Bautista, M. J. (2000). *Modelos de computación flexible para la recuperación de información*. Ph.D. Thesis, Computer Science Faculty, University of Granada.
- Martín-Bautista, M. J., Vila, M. A., & Larsen, H. L. (1999). A fuzzy genetic algorithm approach to an adaptive information retrieval agent. *Journal of the American Society for Information Science*, 50(9), 760–771.
- Michalewicz, Z. (1995). *Genetic algorithms + data structures = evolution programs*. Berlin: Springer-Verlag.
- Noreault, T., McGill, M., & Koll, M. B. (1981). *A performance evaluation of similarity measures, document term weighting schemes and representation in a Boolean environment*. *Information retrieval research*. London: Butterworths.
- Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, 14(3), 130–137.
- Raghavan, V. V., & Agarwal, B. (1987). Optimal determination of user-oriented clusters: an application for the reproductive plan. In *Genetic algorithms and their applications: Proceedings of the second international conference on genetic algorithms and their applications* (pp. 241–246).
- Raghavan, V. V., & Birchard, K. (1979). A clustering strategy based on a formalism of the reproduction process in natural systems. In *Proceedings of the second international ACM/SIGIR conference on information retrieval* (pp. 10–22).
- Robertson, A. M., & Willett, P. (1994). Generation of equipotent groups of words using a genetic algorithm. *Journal of Documentation*, 50(3), 213–232.
- Robertson, A. M., Willett, P. (1995). *Use of genetic algorithms in information retrieval*. British Library, Research and Development Department, BLRD Report 6201.
- Robertson, A. M., & Willett, P. (1996). An upperbound to the performance of ranked-output searching: optimal weighting of query terms using a genetic algorithm. *Journal of Documentation*, 52(4), 405–420.
- Robertson, G. (1987). Parallel implementation of genetic algorithms in classification systems. In L. Davis (Ed.), *Genetic algorithms and simulated annealing* (pp. 129–140). San Mateo, CA: Morgan Kaufmann.
- Salton, G., & Buckley, C. (1988). Parallel text search methods. *Communications of the ACM*, 31, 202–215.
- Salton, G., & Buckley, C. (1990). Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4), 288–297.
- Salton, G., & McGill, M. J. (1983). *Introduction to modern information retrieval*. Cambridge, MA: McGraw-Hill.
- Sanchez, E. (1994). Fuzzy logic and genetic algorithms in information retrieval. In *Proceedings of the third international conference on fuzzy logic, neural networks and soft computing* (pp. 29–35).
- Sanchez, E., Miyano, H., & Brachet, J. (1995). Optimization of fuzzy queries with genetic algorithms. Application to a data base of patents in biomedical engineering. *VI IFSA Congress, Sao Paulo, Brazil* (Vol. II, pp. 293–296).
- Sanchez, E., & Pierre, P. (1994). Fuzzy logic and genetic algorithms in information retrieval. In *Third international conference on fuzzy logic, neural networks and soft computing, Izuaka, Japan* (pp. 29–35).

- Smith, M. P., & Smith, M. (1997). The use of genetic programming to build boolean queries for text retrieval through relevance feedback. *Journal of Information Science*, 23(6), 423–431.
- Vrajitoru, D. (1997). *Apprentissage en recherche d'informations*. Doctoral thesis, Faculty of Science, University of Neuchâtel.
- Vrajitoru, V. (1998). Crossover improvement for the genetic algorithm in information retrieval. *Information Processing and Management*, 34(4), 405–415.
- Yang, J. J., & Korfhage, R. R. (1992). Adaptive information retrieval systems in vector model. In *Proceedings of the symposium on document analysis and information retrieval. Las Vegas, Nevada* (pp. 134–150).
- Yang, J. J., & Korfhage, R. R. (1993). Query optimization in information retrieval using genetic algorithms. In *Proceedings of the fifth international conference on genetic algorithms, Urbana, IL* (pp. 603–611).
- Yang, J. J., & Korfhage, R. (1994). Query modification using genetic algorithms in vector space models. *International Journal of Expert Systems*, 7(2), 165–191.