Using Genetic Algorithm to Improve Information Retrieval Systems

Ahmed A. A. Radwan, Bahgat A. Abdel Latef, Abdel Mgeid A. Ali, and Osman A. Sadek

Abstract—This study investigates the use of genetic algorithms in information retrieval. The method is shown to be applicable to three well-known documents collections, where more relevant documents are presented to users in the genetic modification. In this paper we present a new fitness function for approximate information retrieval which is very fast and very flexible, than cosine similarity fitness function.

Keywords—Cosine similarity, Fitness function, Genetic Algorithm, Information Retrieval, Query learning.

I. INTRODUCTION

GENETIC Algorithm (GA) is a probabilistic algorithm simulating the mechanism of natural selection of living organisms and is often used to solve problems having expensive solutions. In GA, the search space is composed of candidate solutions to the problem, each represented by a string is termed as a chromosome. Each chromosome has an objective function value, called fitness. A set of chromosomes together with their associated fitness is called the population. This population, at a given iteration of the genetic algorithm, is called a generation. Holland, De Jong and Goldberg were pioneered of GA in the context of continuous non-linear optimization [1], [2] and [3].

Genetic algorithms (GAs) are not new to information retrieval [4], [5]. Gordon suggested representing a posting as a chromosome and using genetic algorithms to select good indexes [6]. Yang *et al.* suggested using GAs with user feedback to choose weights for search terms in a query [7]. Morgan and Kilgour suggested an intermediary between the user and IR system employing GAs to choose search terms from a thesaurus and dictionary [8]. Boughanem et al. [9], Horng and Yeh [10], and Vrajitoru [11], examine GAs for information retrieval and they suggested new crossover and mutation operators. Vrajitoru examined the effect of population size on learning ability, concluding that a large population size is important [12].

A. A. A. Radwan is Prof. and the head of Computer Science Department, Minia University (Corresponding author to provide mobile: 0101075033, e-mail: aaaradwaneg@yahoo.co.uk, IEEE member for 5 years).

B. A. Abdel Latef is assistant prof., Department of Computer Science, Minia University (e-mail: dr_bahgat2005@yahoo.com).

A. A. Ali is assistant prof., Department of Computer Science, Minia University (e-mail: abdelmgeid@yahoo.com).

O. A. Sadek is a Demonstrator, Department of Computer Science, Minia University (e-mail: oas_as@yahoo.com).

Despite the successes, little use has been made of genetic algorithms for Ad-Hoc queries. Harman observed different IR systems returning substantially different results, yet maintaining approximately equal performance [13]. Building on the, Bartell *et al.* suggestion, in which we combine the output of different ranking functions to improve performance [14]. Pathak *et al.* used a genetic algorithm to choose weights for such a combination [15].

In this paper we introduce a new fitness function and compare its results with GA based on Cosine fitness function and Classical IR in query learning problems. Our fitness function has been applied on three well-known test collections (CISI, CACM and NPL), to gain an exhaustive view of improvement information retrieval systems using genetic techniques.

II. ANTECEDENTS

A. Information Retrieval

Information Retrieval System (IRS), that is, a system used to store items of information that need to be processed, searched and retrieved corresponding to a user's query. Most IRSs use keywords to retrieve documents. The systems first extract keywords from documents and then assign weights to the keywords by using different approaches. Such a system has two major problems. One is how to extract keywords precisely and the other is how to decide the weight of each keyword. This research presents an application of GA as relevant feedback method aiming to adapt keywords weights. An IRS is basically constituted by three main components, whose composition is introduced as follows [16], [17].

- **The documentary database**. This component stores the documents and the representations of their information contents. It is associated with the indexer module, which automatically generates a representation for each document by extracting the document contents. Textual document representation is typically based on index terms (that can be either single terms or sequences), which are the content identifiers of the documents.

- **The query subsystem**. It allows the users to formulate their information needs and presents the relevant documents retrieved by the system to them. To do that, it includes a query language that collects the rules to generate legitimate queries and procedures to select the relevant documents.

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:5, 2008

- **The matching mechanism**. It evaluates the degree to the document, which representations satisfy the requirements expressed in the query, the Retrieval Status Value (RSV) and retrieves those documents that are judged to be relevant to it.

B. Information Retrieval Models

Several retrieval models have been studied and developed in the IR area; we analyze some of these models, which are:

Boolean model. In the Boolean retrieval model, the indexer module performs a binary indexing in the sense that a term in a document representation is either significant (appears at least once in it) or not. User queries in this model are expressed using a query language that is based on these terms and allows combinations of simple user requirements with the logical operators AND, OR and NOT. The result obtained from the processing of a query is a set of documents that totally match with it, i.e., only two possibilities are considered for each document: to be or not to be relevant for the user's needs, represented by the user query [17], [18].

Vector space model. In this model, a document is viewed as a vector in n-dimensional document space (where n is the number of distinguishing terms used to describe contents of the documents in a collection) and each term represents one dimension in the document space. A query is also treated in the same way and constructed from the terms and weights provided in the user request. Document retrieval is based on the measurement of the similarity between the query and the documents. This means that documents with a higher similarity to the query are judged to be more relevant to it and should be retrieved by the IRS in a higher position in the list of retrieved documents. In This method, the retrieved documents can be orderly presented to the user with respect to their relevance to the query [17].

Probabilistic model. This model tries to use the probability theory to build the search function and its operation mode. The information used to compose the search function is obtained from the distribution of the index terms throughout the collection of documents or a subset of it. This information is used to set the values of some parameters of the search function, which is composed of a set of weights associated to the index terms [19], [20].

C. Evaluation of Information Retrieval Systems

There are several ways to measure the quality of an IRS, such as the system efficiency and effectiveness, and several subjective aspects related to the user satisfaction. Traditionally, the retrieval effectiveness (usually based on the document relevance with respect to the user's needs) is the most considered. There are different criteria to measure this aspect, with the precision and the recall being the most used.

Precision (P) is the rate between the relevant documents retrieved by the IRS in response to a query and the total number of documents retrieved, whilst Recall (R) is the rate

between the number of relevant documents retrieved and the total number of relevant documents to the query existing in the database [18]. The mathematical expression of each of them is showed as follows:

$$P = \frac{\sum_d r_d \cdot f_d}{\sum_d f_d} \quad , \quad R = \frac{\sum_d r_d \cdot f_d}{\sum_d r_d} \longrightarrow \quad (1)$$

with $r_d \in \{0, 1\}$ being the relevance of document d for the user and $f_d \in \{0, 1\}$ being the retrieval of document d in the processing of the current query. Notice that both measures are defined in [0,1], with being the optimal value.

The evaluation function herein is the non-interpolated average precision [21], [22]. Which is similar to average precision but with the cutoff points equivalent to the training documents. In this measure function, the documents are simply ranked. Let d1, d2, . . ., $d_{|D|}$ denote the sorted documents by decreasing order of the values of the similarity measure function, where $|D|$ represents the number of training documents. The function r (d) gives the relevance of a document d. It returns 1 if d is relevant, and 0 otherwise. The non-interpolated average precision is defined as follows:

$$AvgP = \frac{1}{|D|} \sum_{i=1}^{|D|} r(d_i) \cdot \sum_{j=1}^{|D|} \frac{1}{j} \longrightarrow \quad (2)$$

when $r(d_i)$ returns 1, if d_i is relevant and 0 otherwise where $|D|$ represent the number of documents [21].

III. Some Applications of GAs in Information Retrieval

There has been an increasing interest in the application of GA tools to IR in the last few years. Concretely, the machine learning paradigm [23], whose aim is the design of system able to automatically acquire knowledge by themselves, seems to be interesting in this topic [24].

GAs are not specifically learning algorithms, but also offering a powerful and domain independent search ability that can be used in many learning tasks, since learning and self-organization can be considered as optimization problems in many cases. Due to this reason, the applications of GAs to IR have increased in the last decade. Among others, next subsections show some of different proposals made in these areas in the last few years.

A. Automatic Document Indexing

The applications in this area to adapt the descriptions of the documents in the documentary base with the aim of facilitating document retrieval in the face of relevant queries.

Gordon proposes a GA to derive the document descriptions. He chooses a binary coding scheme where each description is

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:5, 2008

a fixed length and a binary vector [28]. The genetic population is composed of different descriptions for the same document.

The fitness function is based on calculating the similarity between the current document description and each of the queries (for which the document is relevant or non-relevant) by means of the Jaccard's index and then computing the average adaptation values of the description for the set of relevant and non-relevant queries.

In Gordon work, GA considered is quite unusual as there is no mutation operator and the crossover probability is equal to 1. With regard to the selection scheme, the number of copies of each chromosome in the new population is calculated and dividing its adaptation value by the population average.

Fan *et al*. propose an algorithm for indexing function learning based on GA, whose aim to obtain an indexing function for the key term weighting of a documentary collection to improve the IR process [25].

B. Clustering of Documents and Terms

In this area, two approaches have been applied for obtaining user-oriented document clusters. Robertson and Willet look for groups of terms appearing with similar frequencies in the documents of a collection [26]. The authors consider a GA grouping the terms without maintaining their initial order. The main features of the GA are:

- **Representation scheme.** Two different coding schemes are considered: separator method and division-assignment method.

- **Initial population.** The first generation of the chromosomes depends on the chosen coding and the rest of individuals are randomly generated.

- **Operators.** Each operator has an application probability associated and it is selected spinning the roulette. Different crossover and mutation operators are used.

- **Fitness function.** There are two proposals:

    o A measure of the relative entropy and

    o Pratt's measure.

C. Matching Function Learning

The aim of matching function learning is to use a GA to generate a similarity measure for a vector space IRS to improve its retrieval efficiency for a specific user. This constitutes a new relevance feedback philosophy since matching functions are adapted instead of queries. Two different variants have been proposed in the specialized literature:

- Linear combination of existing similarity functions. In Pathak *et al.* propose a new weighted matching function, which is the linear combination of different existing similarity functions [15]. The weighting

parameters are estimated by a GA based on relevance feedback from users. They use real coding, a classical generational scheme, two-point crossover and Gaussian noise mutation. The algorithm is tested on the Cranfield collection.

- Automatic similarity measure learning. A GA to automatically learn a matching function with relevance feedback is introduced in [27], [28]. The similarity functions are represented as trees and a classical generational scheme, the usual GA crossover are considered.

D. Query Learning

This is the most extended group of applications of GAs in IR. Every proposal in this group use GAs either like a relevance feedback technique or like an Inductive Query By Example (IQBE) algorithm.

The basis of relevance feedback lies in the fact that either users normally formulate queries composed of terms, which do not match the terms (which used to index the relevant documents to their needs) or they do not provide the appropriate weights for the query terms. The operation mode is involving and modifying the previous query (adding and removing terms or changing the weights of the existing query terms) with taking into account the relevance judgments of the documents retrieved by it, constitutes a good way to solve the latter two problems and to improve the precision, and especially the recall, of the previous query [18].

IQBE was proposed in as "a process in which searchers provide sample documents (examples) and the algorithms induce (or learn) the key concepts in order to find other relevant documents'' [24]. This method is a process for assisting the users in the query formulation process performed by machine learning methods. It works by taking a set of relevant (and optionally, non-relevant documents) provided by a user and applying an off-line learning process to automatically generate a query describing the user's needs.

Smith and Smith propose a GA for learning queries for Boolean IRSs [29]. Although they introduce it as a relevance feedback algorithm, the experimentation is actually closer to the IQBE framework. The algorithm components are described as follows:

- The Boolean queries are encoded in expression trees, whose terminal nodes are query terms and whose inner nodes are the Boolean operators AND, OR and NOT.
- Each generation is based on selecting two parents, with the best fitted having a larger chance to be chosen, and generating two offspring from them. Both offspring are added to the current population which increments its size in this way.
- The usual GA crossover is considered [30]. No mutation operator is applied.
- The initial population is generated by randomly selecting the terms included in the set of relevant documents provided by the user, having those present

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:5, 2008

in more documents a higher probability of being selected.

- The fitness function gives a composite retrieval evaluation encompassing the two main retrieval parameters (precision and recall).

Yang and Korfaghe [31] propose a similar GA to that of Robertson and Willet's [26]. They use a real coding with the two-point crossover and random mutation operators (besides, crossover and mutation probabilities are changed throughout the GA run). The selection is based on a classic generational scheme where the chromosomes with a fitness value below the average of the population are eliminated, and the reproduction is performed by Baker's mechanism.

IV. SYSTEM FRAMEWORK

A. Building IR System

The proposed system is based on Vector Space Model (VSM) in which both documents and queries are represented as vectors. Firstly, to determine documents terms, we used the following procedure:

- Extraction of all the words from each document.

- Elimination of the stop-words from a stop-word list generated with the frequency dictionary of Kucera and Francis [32].

- Stemming the remaining words using the porter stemmer that is the most commonly used stemmer in English [16], [33].

After using this procedure, the final number of terms was 6385 for the CISI collection, 7126 for CACM and 7772 for NPL. After determining the terms that described all documents of the collection, we assigned the weights by using the following formula which proposed by Salton and Buckley [34]:

$$a_{ij} = \frac{\left(0.5 + 0.5\dfrac{tf_{ij}}{\max tf}\right) \times \log \dfrac{N}{n_i}}{\sqrt{\left(0.5 + 0.5\dfrac{tf_{ij}}{\max tf}\right)^2 \times \left(\log \dfrac{N}{n_i}\right)^2}} \longrightarrow \quad (3)$$

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 finally, N is the total number of documents in the database.

Finally, we normalize the vectors, dividing them by their Euclidean norm. This is according to the study of Noreault *et al.*, of the best similarity measures which makes angle comparisons between vectors [35].

We carry out a similar procedure with the collection of queries, thereby obtaining the normalized query vectors. Then, we apply the following steps:

- 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 documents of the collection.

- This list is ranked in decreasing order of similarity degree.

- Make a training data consists of the top 15 document of the list with a corresponding query.

- Automatically, the keywords (terms) are retrieved from the training data and the terms which are used to form a binary query vector.

- Adapt the query vector using the genetic approach.

B. The Genetic Approach

Once significant keywords are extracted from training data (relevant and irrelevant documents) including weights are assigned to the keywords. The binary weights of the keywords are formed as a query vector. We have applied GA for two fitness function to get an optimal or near optimal query vector, also we have compared the result of the two GA approach with the classical IR Systems without using GA. This will be explained in the following subsections.

1) Representation of the chromosomes

These chromosomes use a binary representation, and are converted to a real representation by using a random function. We will have the same number of genes (components) as the query and the feedback documents have terms with non-zero weights. The set of terms contained in these documents and the query is calculated. The size of the chromosomes will be equal to the number of terms of that set, we get the query vector as a binary representation and applying the random function to modify the terms weights to real representation. Our GA approach receives an initial population chromosomes corresponding to the top 15 documents retrieved from classical IR with respect to that query.

2) Fitness function

Fitness function is a performance measure or reward function, which evaluates how each solution, is good. In our work, we used two GAs with two different fitness functions: (a) the first GA system (GA1) uses a measure of cosine similarity between the query vector and the chromosomes of the population as a fitness function, with the equation:

$$\frac{\sum_{i=1}^{t} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{t} x_i^2 \cdot \sum_{i=1}^{t} y_i^2}} \longrightarrow \quad (4)$$

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:5, 2008

where X_i is the real representation weight of term i in the chromosome, Y_i is the real representation weight of that term in the query vector and t is the total number of terms in the query vector as in a given chromosome . The value of the cosine similarity lies on the interval [0,1] according to the similarity between a chromosome and the query.

(b) The second GA2 uses a new fitness function represents by:

$$\sum_{i=1}^{t} x_i - y_i \longrightarrow \quad (5)$$

which is the difference between terms weights of a given chromosome and the query vector.

3) Selection

As the selection mechanism, the GA uses ''simple random sampling'' [1], [3]. This consists of 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. Also, we have used the ''elitism'' strategy, as a complement to the selection mechanism [2]. After generating the new population, if 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.

4) Operators

In our GA approaches, we use two GA operators to produce offspring chromosomes, which are:

Crossover is the genetic operator that mixes two chromosomes together to form new offspring. Crossover occurs only with crossover probability Pc. Chromosomes are not subjected to crossover remain unmodified. The intuition behind crossover is exploration of a new solutions and exploitation of old solutions. GAs construct a better solution by mixture good characteristic of chromosome together. Higher fitness chromosome has an opportunity to be selected more than lower ones, so good solution always alive to the next generation. We use a single point crossover, exchanges the weights of sub-vector between two chromosomes, which are candidate for this process.

Mutation is the second operator uses in our GA systems. Mutation involves the modification of the gene values of a solution with some probability Pm. In accordance with changing some bit values of chromosomes give the different breeds. Chromosome may be better or poorer than old chromosome. If they are poorer than old chromosome they are eliminated in selection step. The objective of mutation is restoring lost and exploring variety of data.

V. EXPERIMENTAL RESULTS

The test databases for our GA approaches are three well-known test collections, which are: the CISI collection (1460

documents on information science), the CACM collection (3204 documents on Communications), and finally the NPL collection (11,429 documents on electronic engineering). One of the principal reasons for choosing more than one test collection is to emphasize and generalize our results in all alternative test documents collections. We apply the Experiments on 100 queries and we choose these queries according to each query do not retrieve 15 relevant documents for our IR system. From our experimental observation, the best values for this test documents collections at crossover probability Pc = 0.8 and mutation rate is Pm= 0.7 for the two GAs (GA1 and GA2).

In the following subsections, the results for applying GAs for 100 generation for each GA are explained.

A. The CISI Documents Collection

The results for the two GAs (see Table I), by using non-interpolated average Recall – Precision relationship. From this table we notice that GA2 gives a high improvement than GA1 with 2% and both higher than classic IR system with 13.6% and 11.9%, respectively as average values. Also, the average number of terms of query vector before applying GAs is 509.61 terms, these terms are reduced after applying GA1 to 358.84 terms, and reduced after using GA2 to 83.7 terms.

TABLE I
THE EXPERIMENTAL RESULTS OF CISI COLLECTION

Average Recall Precision for 100 query in CISI Collection					
Recall	Precision			GA1 Improvement %	GA2 Improvement %
	Classic IR	GA1	GA2		
0.1	0.679	0.877	0.88	29.1	29.6
0.2	0.558	0.658	0.66	18	18.36
0.3	0.462	0.585	0.592	26.52	28.12
0.4	0.401	0.4442	0.452	10.8	12.7
0.5	0.3494	0.404	0.4065	15.5	16.3
0.6	0.304	0.311	0.32	2.22	3.89
0.7	0.252	0.265	0.269	5.13	6.84
0.8	0.1989	0.1922	0.199	-3.34	-0.18
0.9	0.1491	0.1538	0.1586	3.18	6.37
Average	0.373	0.4321	0.437	11.91	13.56

B. The NPL Documents Collection

The results for this experiment (see Table II), by using non-interpolated average Recall – Precision relationship. From this table we find that the GA2 gives a high improvement than GA1 with 7.5% and both higher than classic IR system with 19.06% and 11.5%, respectively as average values. Also, the average number of terms of query vector before applying GAs is 134.14 terms; these terms are reduced after applying GA1 to 16.8 terms, and reduced after using GA2 to 21.6 terms.

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:5, 2008

TABLE II
THE EXPERIEMENTAL RESULTS OF NPL COLLECTION

Average Recall Precision for 100 query in NPL Collection					
Recall	Precision			GA1 Improvement %	GA2 Improvement %
	Classic IR	GA1	GA2		
0.1	0.733	0.803	0.803	9.5058	9.5058
0.2	0.5034	0.551	0.561	9.4344	11.39
0.3	0.4351	0.477	0.495	9.534	13.71
0.4	0.3405	0.3857	0.406	13.3	19.3
0.5	0.3133	0.364	0.385	16.135	22.79
0.6	0.24	0.275	0.296	14.44	23.42
0.7	0.2154	0.239	0.261	10.83	21.11
0.8	0.1743	0.194	0.22	11.071	24.187
0.9	0.146	0.1591	0.184	9.317	26.143
Average	0.344	0.383	0.401	11.5065	19.057

C. The CACM Documents Collection

The results for this experiment (see Table III), by using non-interpolated average Recall – Precision relationship.

TABLE III
THE EXPERIEMENTAL RESULTS OF CACM COLLECTION

Average Recall Precision for 100 query in CACM Collection					
Recall	Precision			GA1 Improvement %	GA2 Improvement %
	Classic IR	GA1	GA2		
0.1	0.7227	0.7885	0.808	9.10393	11.8056
0.2	0.4165	0.4738	0.51	13.763	22.383
0.3	0.3694	0.4194	0.453	13.5492	22.556
0.4	0.247	0.274	0.312	11.0097	26.4733
0.5	0.2127	0.225	0.26	5.5784	23.741
0.6	0.158	0.162	0.203	2.741	28.55
0.7	0.1429	0.1397	0.18	-2.248	26.11
0.8	0.1073	0.105	0.15	-2.1804	37.28
0.9	0.09	0.0851	0.128	-5.103	42.591
Average	0.274	0.297	0.334	5.13491	26.832

From this table we notice that GA2 gives a high improvement than we get GA1 with 21.7% and both higher than classic IR system with 26.8% and 5.13%, respectively as average values. Also, the average number of terms of query vector before applying GAs is 160.7 terms; these terms are reduced after applying GA1 to 16.83 terms, and reduced after applying GA2 to 31.5 terms.

VI. CONCLUSION

From pervious results, we note that our new fitness function which is represented by equation (5) gives more sophisticated results than a cosine fitness function in our test collections.

Also, it is easy to prove that the new fitness function has a complexity of order (n^2) while the complexity of the second fitness function (cosine similarity) of order (n^5) for each chromosome, where n is the number of terms in the search space for that query. Also, from the previous tables, we note that: our new fitness function has a precision value better than in cosine similarity fitness function.

The experiments developed use three of the relative document collections (CACM, CISI and NPL), and compare the results of two variant algorithms (Classical IR and GA1) with our fitness function (GA2). The latter algorithm achieves the best performance and it obtains better precision than the other two approach.

REFERENCES

[1] J. H. Holland, "Adaptation in Natural and Artificial Systems", University of Michigan Press, Ann Arbor, 1975.
[2] K. A. DeJong, An Analysis of the Behavior of a Class of Genetic Adaptive Systems, Ph.D. Thesis, University of Michigan, 1975.
[3] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA., 1989.
[4] H. Chen, "Machine learning for information retrieval: neural networks, symbolic learning, and genetic algorithms". Journal of the American Society for Information Science, 46(3), 1995, pp. 194–216.
[5] J. Savoy and D. Vrajitoru, "Evaluation of learning schemes used in information retrieval (CR-I-95-02)". Universite de Neuchatel, Faculte de droit et des Sciences Economiques, 1996.
[6] M. Gordon, "Probabilistic and genetic algorithms in document retrieval". Communications of the ACM, 31(10), 1988, pp. 1208–1218.
[7] J. Yang, R. Korfhage and E. Rasmussen. "Query improvement in information retrieval using genetic algorithms—a report on the experiments of the TREC project". In Proceedings of the 1st text retrieval conference (TREC-1), 1992, pp. 31–58.
[8] J. Morgan and A. Kilgour. "Personalising on-line information retrieval support with a genetic algorithm". In A. Moscardini, & P. Smith (Eds.), PolyModel 16: *Applications of artificial intelligence*, 1996, pp. 142–149.
[9] M. Boughanem, C. Chrisment, and L. Tamine. "On using genetic algorithms for multimodal relevance optimization in information retrieval". Journal of the American Society for Information Science and Technology, 53(11), 2002, pp. 934–942.
[10] J. T. Horng and C. C. Yeh. "Applying genetic algorithms to query optimization in document retrieval". Information Processing & Management, 36(5), 2000, pp. 737–759.
[11] D. Vrajitoru. "Crossover improvement for the genetic algorithm in information retrieval". Information Processing& Management, 34(4), 1998, pp. 405–415.
[12] D. Vrajitoru. "Large population or many generations for genetic algorithms? Implications in information retrieval". In F. Crestani and G. Pasi (Eds.), *Soft computing in information retrieval. Techniques and applications*, Physica-Verlag, 2000, pp. 199–222.
[13] D. Harman. "Overview of the first TREC conference". In Proceedings of the *16th ACM SIGIR conference on information retrieval*, 1993, pp. 36–47.
[14] B. T. Bartell, G. W. Cottrell and R. K. Belew. "Automatic combination of multiple ranked retrieval systems". In Proceedings of the *17th ACM SIGIR conference on information retrieval*, 1994, pp. 173–181.
[15] P. Pathak, M. Gordon and W. Fan. "Effective information retrieval using genetic algorithms based matching functions adaption", in: Proc. *33rd Hawaii International Conference on Science (HICS), Hawaii*, USA, 2000.
[16] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*, Adisson, 1999.
[17] G. Salton and M.H. McGill. *Introduction to Modern Information Retrieval*, McGraw-Hill, 1983.
[18] C.J. Van Rijsbergen. *Information Retrieval*, second ed., Butterworth, 1979.
[19] A. Bookstein. "Outline of a general probabilistic retrieval model", Journal of Documentation 39 (2), 1983, pp. 63–72.

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:5, 2008

[20] N. Fuhr. "Probabilistic models in information retrieval", Computer Journal 35 (3), 1992, pp. 243–255.

[21] C. H. Chang and C. C. Hsu. *The design of an information system for hypertext retrieval and automatic discovery on WWW. Ph.D. thesis*, Department of CSIE, National Taiwan University, 1999.

[22] K. L. Kwok. "Comparing representations in Chinese information retrieval". *ACM SIGIR'97*, Philadelphia, PA, USA, 1997, pp. 34 -41.

[23] T. Mitchell. *Machine Learning*, McGraw-Hill, 1997.

[24] H. Chen et al., "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), 1998, pp. 693–705.

[25] W. Fan, M.D. Gordon and P. Pathak. "Personalization of search engine services for effective retrieval and knowledge management", in: Proc. 2000 *International Conference on Information Systems (ICIS)*, Brisbane, Australia, 2000.

[26] A.M. Robertson and P. Willet. "Generation of equifrequent groups of words using a genetic algorithm", Journal of Documentation 50 (3), 1994, pp. 213–232.

[27] M. Gordon. "Probabilistic and genetic algorithms for document retrieval", Communications of the ACM 31 (10), 1988, pp. 1208–1218.

[28] W. Fan, M.D. Gordon and P. Pathak. "Discovery of context-specific ranking functions for effective information retrieval using genetic programming", *IEEE Transactions on knowledge and Data Engineering, in press.*

[29] M.P. Smith, M. Smith. "The use of genetic programming to build Boolean queries for text retrieval through relevance feedback", Journal of Information Science 23 (6), 1997, pp. 423–431.

[30] J. Koza. "Genetic Programming". *On the Programming of Computers by means of Natural Selection*, The MIT Press, 1992.

[31] J. Yang and R. Korfhage. "Query modifications using genetic algorithms in vector space models", *International Journal of Expert Systems* 7 (2), 1994, pp.165–191.

[32] H. Kucera and N. Francis. "Computational analysis of present-day American English". Providence, RD: Brown University Press, 1967.

[33] M. F. Porter. "An algorithm for suffix stripping. Program", 14(3), 1980, pp. 130–137.

[34] G. Salton and C. Buckley. "Improving retrieval performance by relevance feedback". Journal of the American Society for Information Science, 41(4), 1990, pp. 288–297.

[35] T. Noreault, M. McGill and M. B. Koll. "A performance evaluation of similarity measures, document term weighting schemes and representation in a Boolean environment". Information retrieval research. London: Butterworths, 1981.